Advanced Civ

Mod for BtS 3.19   version 1.04   23 Dec 2021

by Georg W. (firpo)

User Manual

AdvCiv is a mod for advanced players, building on karadoc's "K-Mod: Far Beyond the Sword." AdvCiv does not add content, and makes only few notable balance changes apart from changes to the AI. Multiplayer (except PitBoss and PBEM) is supported but not thoroughly tested. For questions, comments and current activities, please visit the development [thread](#) on CivFanatics.

Installation

To install AdvCiv, copy the unzipped AdvCiv folder into `Beyond the Sword\Mods` under `Program Files`. When the mod is loaded for the first time, it creates a folder `My Games\ Beyond the Sword\AdvCiv` for the [BUG](#) settings. You'll have to remove this folder manually to fully uninstall AdvCiv.

The above assumes a BtS installation from optical disc on Windows 8 or earlier, though I understand that it works the same way with the GOG.com edition and on Windows 10. Steam users may have to [revert](#) to the Game Spy version in order to run any mods, and the `Mods` folder should be in `Steamapps\Common\Sid Meier's Civilization IV Beyond the Sword\Beyond the Sword`. On Unix-based systems, the mod can only be played through a virtual machine or Wine. (These guides might help, but I haven't tried either of them: [MacOS](#), [Linux](#). [This](#) CFC post also looks pertinent.)

To start the mod through a Windows shortcut, create a shortcut to `Civ4BeyondSword.exe`, open the shortcut's file properties, and add a space and `mod=\AdvCiv` to the `Target` field.

Contents

To start playing, it should suffice to read this overview of the mod's original content and included third-party components. The subsequent chapters describe the major changes in more detail. Most of the AI and UI changes are low-key though, and only covered by the (extensive) appendix. I've assigned a numeric id to each batch of changes; when an id is mentioned, there's always a link to further details and rationales in the appendix.

- UWAI: Utility-Based War AI. Rewrite of the AI parts dealing with decisions on war and peace. The new AI is better at anticipating the course of a war. Conditions that rule out war categorically in BtS (e.g. attitude or being occupied with another war) are factored into an overall utility value and can be outweighed by factors that favor war. UWAI can be disabled through the "Aggressive AI" game option.

- Dynamic Diplomacy: Makes several relations bonuses harder to get, particularly "mutual struggle," "open borders" and "supplied resources." All AI memory decays, including "You declared war on us." Two new modifiers: "We oppose your ruthless expansionism" and "You agreed to come to our aid in wartime". +4 relations is required for "pleased" attitude and -1 for "cautious," but this is evened out by a change to the "first impression" modifier. The AI will (sometimes) offer and accept cities in trade.

- Revised Barbarians: Barbarian activity increases more gradually; fewer Barbarians on low-yield tiles; creation rate adjusted to game speed; fogbusting nerfed (units can appear on all fogged tiles); land units can spawn aboard Galleys; Great Wall reworked; goodies from Tribal Villages adjusted to game progress; Scout and Explorer buffed.

- Immortal Culture: Culture of defeated civs stays in the game and can cause anger and revolts. Revolts can occur in any cities, not just near a border. Shortened occupation countdown in conquered cities, but the countdown decreases only probabilistically based on the strength of the occupying force.

- K-Mod v1.46 by karadoc: substantial AI improvements (incl. BBAI); usability improvements (incl. BUG, which in turn includes Civ4lerts, reminder, Sevopedia and several optional advisor screens); bugfixes (incl. the unofficial patches); performance tweaks; game balance tweaks (several from PIG, Lead From Behind); practically no flavor changes other than Actual Quotes. The main author of BBAI is jdog5000. For BUG credits, see the BUG help file (Alt+Ctrl+F1 in a running game).

- Kek-Mod v0.25b, a K-Mod fork by DarkLunaPhantom: partly included (mostly bugfixes).

- Show Hidden Attitude Mod by DaveMcW

- Various bugfixes, performance optimizations and UI and AI tweaks adopted from: We the People, BULL, More Naval AI, Caveman2Cosmos, Rise of Mankind, Dawn of Civilization

- Various changes, especially to rules and AI behaviors that are easy to take advantage of in BtS; e.g. can't steal Workers.

- R&F: Rise and Fall. Game option that divides the game into chapters; you take control of a different civ in each chapter. A score is awarded after each chapter based on how the standing of your civ has changed.

- SPaH: Start Points as Handicap. Game option that gives an Advanced Start to the AI civs but not to the human civs. Start points can also be assigned unequally to the AI civs; the mod then places the civs with the most points the farthest away from the human civs initially.

- [PerfectMongoose](#) map script (v3.3; main authors: Cephalo and LunarMongoose)
- [Mixed Continents](#) map script based on jam3's Islands and Continents v1.1.
- [savemap](#) function by tywiggins and xyx
- Blue Marble "light": Altered terrain textures that are easier on the eyes; based on Kai Fiebach's [Blue Marble](#), but closer to the original textures. (How to disable: see [appendix](#).)

Utility-Based War AI

UWAI is enabled by default, but players who prefer the K-Mod AI, can use the "Aggressive AI (Legacy)" option on the Custom Game screen to disable UWAI; that option also enables the Aggressive AI mode. There aren't separate options for UWAI and Aggressive AI because UWAI doesn't have an aggressive or non-aggressive mode (see also change [019](#); K-Mod without Aggressive AI can still be configured in `GlobalDefines_advc.xml`).

The major differences between UWAI and BtS/K-Mod:

UWAI	*BtS/K-Mod*
Military analysis: UWAI predicts how ongoing and hypothetical wars might develop in the medium-term, i.e. over the next 25 to 50 turns. The projection is based on power ratings for various military branches, namely the army (land and air units available for offensives), home guard (only available for defense), fleet (sea units), logistics (cargo ships) and nukes. Simulates military build-up (based on estimated per-turn production), clashes of army stacks, naval landings and attempted conquests of specific cities. The outcomes include cities lost and conquered by the various war parties, invested production and lost military power. These predictions aren't intended to be accurate; they only need to be plausible enough to result in plausible AI behavior.	The BtS military analysis simply compares present power ratings. Each civ has a single power rating, i.e. land and sea units aren't distinguished. K-Mod adds a projection of military build-up.
Power: Power ratings are based on per-unit military power values that are, essentially, computed as combat strength raised to the power of 1.66. E.g. Swordsman has about 20 power and Cuirassier 75; this value also accounts for the faster movement and other useful abilities of Cuirassiers. Caveat: Many other AI decisions and the power graph on the Info screen are still based on BtS power ratings.	The power value of a unit normally equals its combat strength; e.g. two Swordsmen count as much as one Cuirassier. That is, BtS underestimates the power of technologically advanced units.

The **evaluation** of the expected military outcomes is broken down into 27 aspects, each dealing with one specific reason for war (e.g. gaining more cities or loathing for a war opponent) or against (e.g. diplomatic ill will or invested production). The result of the evaluation is a utility value indicating how worthwhile the war (plan) in question is.	The AI considers attitude, military power, geographical proximity and victory strategies. These factors are combined in an arcane manner by a function named `startWarVal`. Peace is evaluated separately by the `endWarVal` function, which is based on war successes, finances and the tactical situation. K-Mod has expanded `startWarVal`, but it's still inconsistent with `endWarVal`, leading to war-peace oscillation, and too ad hoc for this important part of the AI.
War plans are fully **re-evaluated** each turn while in preparation, and canceled if they no longer appear worthwhile.	War plans in preparation are only canceled in rare circumstances.
The only necessary condition for beginning (and continuing) war preparations is a **positive war utility** value. Attitude and power are covered by the war utility calculation and no longer work as hard requirements. That said, attitude has a particularly strong impact on war utility, so that a relatively peaceful leader like Saladin doesn't go to war against a civ that he is pleased with unless there are unusually strong reasons for the war. Once preparations are concluded and war is imminent, it's too late for a change in attitude to stop the war.	Thresholds for attitude and power ratio serve as necessary conditions for war; in particular, many AI leaders never start war preparations against civs they're pleased with. (Exceptions: Aggressive AI option and – in BBAI/ K-Mod – AI leaders close to a military victory.)
If war utility is positive, the AI begins war preparations with a per-turn probability based on personality and the utility value. As a result, the AI can be **quick to declare war** in response to another declaration of war or some rapid shift in power (e.g. after upgrading to Riflemen).	The per-turn probability is based only on personality and attitude.
The AI **makes peace** if and only if war utility is negative or outweighed by reparations.	Apart from an `endWarVal` check, there are some special conditions for peace, such as having fought to a long draw or being alone in what had started as a dogpile war.

After a declaration of war, the AI **refuses to talk** for just one turn. After that, the AI can be contacted unless the price for peace is greater than what the other side is able to pay. Exception: If war was declared at the request of a third party or through a defensive pact or a vote, the attacked AI civ refuses to talk to its war enemies for the same duration as in BtS.	The refuse-to-talk duration is based on AI personality and war success. Possible reparations don't play a role.
The AI can start a war **while already in another war**. It's possible, in principle, to hire an AI civ for war even if that AI civ is already preparing or fighting a war, but the price is usually prohibitive.	The AI doesn't consider war preparations when it's already in a war; K-Mod relaxes this restriction in the endgame. In both BtS and K-Mod, the AI can't be hired while already conducting or planning war.
The AI refuses requests for starting a war regardless of war utility if either its attitude toward the proposed target is too high, or its attitude toward the sponsor is too low. Otherwise, the necessary payment is computed based on war utility. If the result exceeds a threshold, the AI refuses to declare war, stating "We are afraid of their military might" if the power ratio is highly unfavorable, and "**We have enough on our hands right now**" otherwise. Thus, "enough on our hands" doesn't imply that the AI is already preparing another war. When an AI leader becomes willing to discuss a sponsored war, the player is notified through an alert message (change [210a]) and the Glance tab of the Foreign Advisor screen shows a fist icon ([152]).	Same: "We couldn't betray close friends" or "We don't like you enough." The AI also refuses categorically if the power ratio is too unfavorable or when already preparing or fighting a war. In BtS, if the trade screen says "enough on our hands," then the AI is definitely preparing a war. In K-Mod, other reasons for not wanting to start a war can block the enough-on-hands tell. For the player's convenience, the BUG mod shows a fist icon on the scoreboard next to any AI civ that will give the "enough on our hand" response. (UWAI disables this.)
While the utility of a war plan against a human player is positive but small, and the AI hasn't begun war preparations yet, there's an increased chance of demands for **tribute** or a gift.	Tribute demands are made randomly against any disliked and militarily weaker human civ. Whether the AI is currently ready to start a war doesn't play a role.
UWAI continues to use almost all of the AI **personality** values from BtS. Many work differently but, in most cases, to a similar effect. For example, a high `MaxWarNearbyPowerRatio` leads to an optimistically biased military analysis, which can make an AI leader willing to start wars that he or she may well lose.	Each AI leader has some 20 personality values affecting war and peace, set in `Civ4LeaderHeadInfos.xml`. `MaxWarNearbyPowerRatio` is a threshold that rules out war if the military power ratio is too unfavorable.

Hiring war allies, granting or denying tribute, UN peacekeeping and some **other decisions** that imply war or peace are made based on war utility.	Separate heuristics for these decisions, based on some combination of power ratio and attitude.
Decisions about vassal agreements are still largely made by the K-Mod AI (with many adjustments; change 112), and defensive pacts are still mainly a matter of attitude values.	
Unless a city is clearly about to fall, the AI is *less* inclined to make peace when there are **enemy units near** its cities.	The AI is *more* willing to make peace when there are hostile units near its cities – but can't evaluate if these units actually pose a threat. This can be exploited for better peace deals.
This does not apply to capitulation; the AI capitulates only when faced with a threatening number of hostile units inside its borders (change 112b).	
This is the only part of UWAI that cares about positions of units on the map. UWAI does not control *how* war is conducted, only *if*.	Positions of units have no bearing on whether the AI is willing to capitulate.

Dynamic Diplomacy

One easily noticeable change is in the mapping from relations values to AI attitude:

Relations	≥ 9	≥ 4	≤ 3	≤ -2	≤ -9
AI attitude	Friendly	Pleased	Cautious	Annoyed	Furious

For comparison, in BtS, Cautious begins at -2, Pleased at +3 and Friendly at +10. However, AdvCiv also removes the (hidden) -1 relations penalty on Noble difficulty and above, meaning that, in effect, the thresholds for Pleased and Cautious stay the same and the threshold for Friendly is two lower than in BtS. Change id: 148.

The following changes to relations modifiers should make them more sensible, more dynamic and less exploitable. Changes to vassal agreements are described at the end of this chapter.

– "Our mutual military struggle ...": A bonus of more than 1 now requires getting involved in the war, that is, to inflict or incur losses. The bonus decays over time and is reduced when there is a non-mutual war. (In BtS, it's sufficient to be formally at war, and, once accumulated, the bonus stays for the rest of the game.) Fighting inside an ally's borders is especially effective at increasing the diplo bonus. 130m

– "You agreed to come to our aid in wartime": Joining a war at the request of an AI civ results in a +1 relations bonus that is remembered for 100 turns on average. 130s

– When a war ally is brought in, the two allies automatically sign a 10-turn peace treaty. 146

– "Years of peace" only start to count once a civ is met. 130a

– "You stopped trading with us": One cannot propose an embargo against one's own trade partner; will first have to cancel those trades. The target of the embargo refuses to talk for (on average) 18 turns (BtS: 30) with both the civ that proposed the embargo and

the one that agreed to it. (Exception: when a master asks its vassal to stop trading.) The embargo terminates even recent deals that couldn't otherwise be canceled. [130f](#)

- "You made an arrogant demand" now only applies when a tribute demand is granted; no diplo penalty if the AI refuses to pay. However, if a player declares war on an AI civ that remembers having paid tribute, that civ and all AI civs it has met refuse tribute and help requests by the player for (on average) 80 turns. When an AI civ declares war, it forgets all tribute demands, i.e. it can be attacked without repercussions. [130o](#)

- "You press us too hard": There is always a chance that the AI refuses a request for a gift for no particular reason. A refused request is therefore not a reliable way to determine if the AI is preparing war against the player who makes the request. The AI is willing to grant a gift about every 30 turns (20 in BtS). [144](#)

- "You're getting ahead of us": AI civs dislike civs that are ranked slightly higher on the scoreboard but not those ranked much higher. These rank-based modifiers work differently in BtS (and BtS hides them from the player). [130c](#)

- "A first impression is a lasting one": Another hidden BtS modifier; this one remains constant for the entire game. It is based on leader personalities and affects mostly relations between AI civs. AdvCiv reduces the modifier a bit, specifically the impact of "peace weight," to make diplomacy less preordained. [130b](#)

- "We oppose your ruthless expansionism": Having a high number of cities with foreign majority culture results in relations penalties from everyone. These penalties complement the ones for having vassals ("our rivals being vassals to your empire"). [130w](#)

- "We would have nothing to gain": AI leaders are reluctant to sign Open Borders with civs whom they haven't located yet (not a single revealed land tile) or can't reach. When pleased, most AI leaders sign Open Borders regardless of accessibility. [124](#)

- "Our Open Borders have brought our peoples together": The time that it takes to attain and increase this bonus depends on the distances and trade routes between the two peoples' cities. [130i](#)

- "All the years you have supplied us with resources": This can take many, many years if the recipient already has many resources. [149](#)

- All AI memory decays. For example, memory about how "you razed our cities" decays by one city every 75 turns on average ([130r](#)). The other decay times (no decay in BtS):

Memory type	Turns		Memory type	Turns
"You razed our cities"	75		"You nuked us"	120
"You razed a holy city"	150		"You nuked our friends"	80
"Your spy was caught"	40		"You negotiated a trade embargo"	60
"You made an arrogant demand"	30		"You declared war on our friends"	120
"You liberated our cities"	150		"You brought in a war ally"	75
"You granted us independence"	300		"You declared war on us"	150

- "You declared war on us" penalty reduced to -2 if the declaration of war is triggered by a defensive pact. [130y](#)

- "We are upset that you have signed Defensive Pacts with our rivals": Not if we like these rivals, or have our own defensive pacts with them, or if we're too weak to attack

them anyway. Also no penalty for voluntary vassals ("our rivals being vassals to your empire") if we like those vassals. 130t

–   The AI refuses to sign a Defensive Pact if it was recently canceled (same mechanism as for canceled Open Borders). Defensive Pacts aren't automatically canceled when triggered, i.e. when a third party declares war on one of the signatories. 130p/ kekm.3

–   "We care for our brothers and sisters of the faith"/ "You have wisely chosen your civics": Reduced the upper limit of those modifiers by 1 for about half of the AI leaders (those with the highest limits). If many civs share a religion or civic (e.g. Hereditary Rule), the relations bonuses are reduced. Conversely, the AI is more tolerant toward another religion if either religion is shared by few other civs. 130n

–   "You accepted our state religion/ favorite civic": The bonus decays faster when the player switches out of that religion or civic. 145

–   "Our trade relations have been fair and forthright": No longer based (primarily) on how recently a civ was met; more difficult to max out; and trades that are indeed "fair" contribute to the bonus. (In BtS, only what the AI gets on top of a fair deal counts.) 130p

–   "You have traded with our worst enemy": The penalty is reduced when the worst enemy changes. Trade with a war enemy counts (partially) even if that enemy is not *the worst* enemy. Open Borders contribute to the enemy trade penalty. 130p

    To become a worst enemy at peacetime, the relations value needs to be -3 or worse, i.e. Annoyed attitude isn't necessarily low enough. 148

–   "Perhaps it is time for all this bloodshed to end": The AI sometimes contacts human players with peace offers that can be more generous than those resulting from "what is the price for peace/ stop this fighting". Rejecting such an offer makes the AI slightly less willing to accept human peace offers. (In BtS, a bug prevents AI peace offers.) 134a

–   The AI forgets actions of other civs, like "You gave us help", a little less randomly, whereas the tracking of e.g. "years of peace" or supplied resources is now a bit randomized. (In BtS, e.g. "years of peace" pay off after exactly 60 turns.) 130k/ 130j

AdvCiv eliminates most of the diplomatic drawbacks associated with **capitulated vassals** (change 130v). They have Friendly attitude toward the master, share the master's attitude toward rivals (but no better than Cautious), can't be anyone's worst enemy and are ineligible as war targets and for elections (014). The master civ is held responsible for grievances caused by its capitulated vassals – penalties for razed cities, border troubles and trades with a worst enemy are partially added to the master's relations modifiers –, but the master is not blamed for things that the vassal civ did prior to the vassal agreement, nor for the vassal's religion.

Peacefully acquired vassals still work as in BtS, i.e. as largely independent. The conditions under which the AI agrees to a vassal agreement have been tweaked: Civs only look for a master when feeling threatened – having fallen behind is not reason enough (112). If a master civ fails to protect its voluntary vassal (substantial loss of territory or nuked repeatedly), the vassal cancels the agreement (143). A vassal that gets close to a victory condition also cancels the vassal agreement (112).

The following changes (130h) should make it easier to have productive relations with former vassals after **helping them break free**: Civs in a vassal-master relationship don't consider each other "friends" as far as the "you declared war on our friends" penalty is

concerned. When a vassal makes peace or breaks free, its memory about past declarations of war is decreased. When a capitulated vassal is freed because its master capitulates, the freed vassal gives +2 "You granted us independence" to the master of its former master. Freed vassals and their liberator automatically cease fire. 130y

Happiness from vassals ("We influence other civilizations") is now only gained from voluntary vassals, and it's capped at +1. 142

City trades with the AI are possible under the following conditions:
- The recipient needs to have at least 10% nationality (city tile culture) and at least half as much as the current owner. Exceptions:
  - When negotiating peace, 10% nationality of the recipient is always sufficient.
  - Liberation to a colonial vassal is allowed regardless of nationality.
  - Master to vassal: The vassal needs to have higher nationality than its master.
  - Vassals can never trade or liberate cities to rivals.
- War with the previous owner or nearby enemy units prevent (non-liberation) city trades.
- As in K-Mod, unrevealed cities are treated as secret and thus cannot be traded.
- AI attitude threshold for giving away cities in (non-liberation) trade: Personality-based. For many AI leaders (much) less strict if the owner has less than 20% nationality.
- The AI doesn't give away cities of major economic importance.
- After a (non-liberation) city trade, a peace treaty is signed automatically. Rarely, the AI may reject a city ("we have our reasons") in order to avoid a peace treaty.
- The AI accepts liberated cities only for free, as payment for making peace or in exchange for another liberated city. As in BtS, liberation to the AI improves relations, and only former owners and colonial vassals can receive a city through liberation.

The above doesn't cover all AI trade denial conditions. Full documentation: rules | AI | UI

User interface: The Foreign Advisor screen lists possible city trades on a new tab "Cities", and there is an alert for new city trades (can be disabled on the BUG menu).

Revised Barbarians

(For details, see change id 300 when no other id is stated below.)

- The Barbarian creation rate **increases gradually**, reaching its peak when slightly more than half of the land is claimed by cultural borders, i.e. typically toward the end of the Classical era. The rate is adjusted to the **game speed** setting.

- Fewer Barbarians on **low-yield tiles**: Barbarian units can only be created on habitable tiles (positive food yield when ignoring hills). They're less likely to appear on jungle and tundra without resources, and more likely on all other tiles. The maximal number of Barbarians to be placed per continent takes into account the length of the coastline and disregards nonhabitable tiles; i.e. fewer Barbarians on continents with large (polar) deserts, more on snaky continents. The aim of these changes is to help civs that are surrounded by poor land, and to disadvantage civs that have much good land to settle.

- Barbarian land units can be created **aboard Galleys** in the fog of war. Once a Barbarian Galley has cargo, it moves toward a nearby city for a naval assault. The units can also be dropped along the way, randomly, or to attack an unprotected non-combat unit. The intention is to make Barbarian Galleys harder to ignore, especially for civs without seafood at stake. (change 306)

- "**Fogbusting**" weakened: So long as a tile is not visible to any civ, Barbarian units can

be created there, i.e. the mod disables a BtS rule that had blocked Barbarian placement in a 5x5 square around each civ unit. There are also several changes that make it more appealing to fight Barbarians (rather than ensure that none are created); see below.

– On Monarch, Emperor and Immortal difficulty, human units receive a **+5% combat** modifier against Barbarian units (none in BtS above Prince difficulty). 313

– Barbarian **Spearmen** no longer appear earlier than Axemen. This makes Chariot more useful as an anti-Barbarian unit. 301

– Half of the XP gained from Barbarians (rounded down) counts as **Great General points**. 25% less XP than in BtS is gained from attacks on Barbarians. As in BtS, units with 10 or more XP gain no XP from fighting Barbarians. These restrictions are aimed at "XP farming" tactics that deliberately leave tiles unobserved. 312

– The **Great Wall** (TGW) has two out of the three following effects depending on the Raging Barbarians (RB) and No Barbarians (NB) game options: 310
  ◦ +1 trade route in cities on the same continent except when playing with RB;
  ◦ prevents Barbarians from entering your borders on this continent except with NB;
  ◦ +100% emergence of Great Generals inside your cultural borders if RB or NB.

  To match the flavor of the new trade route effect, TGW switches its Great Person type with Versailles, i.e. TGW generates **Great Merchant** points and Versailles generates Great Spy points. TGW costs **300 production** instead of 150, **requires Archery** and **two Walls** (i.e. Masonry is still required indirectly) in arbitrary cities. It goes **obsolete with Corporation.** As for the increased cost, bear in mind that Barbarians don't become a major threat until the Classical era in AdvCiv.

– Effects from **Tribal Villages** are based on the game turn number. In particular, they will often only yield progress toward a tech rather than the whole tech. Tribal Villages that aren't reached until the midgame may yield several hundred gold or research points (toward any pre-Industrial tech). The other outcomes also become somewhat more potent over the course of a game, may e.g. yield a Worker instead of a Scout. 314

– **Explorers** get the ability to attack Barbarians, which allows them to enter guarded Tribal Villages, but they can't capture cities and no longer receive any free promotions. 315b

– **Scouts** may attack animals (315c) and Scouts get a combat bonus against all Barbarians (315a), not just against animals. On Emperor difficulty and Immortal, the AI starts with one free Scout instead of two and, on Immortal and Deity, with one fewer Archer than in BtS; this leaves more Tribal Villages for human Scouts (250e).

– On continents without any civs, Barbarian cities begin to appear earlier than they do in BtS. This way, a patchy network of Barbarian cities tends to be cover the **New World** by the time it is discovered, reminiscent of Sid Meier's Colonization. So long as these cities outnumber those of all civs taken together, Barbarian units act relatively peacefully.

Immortal Culture

– Eliminating a civ no longer removes that civ's tile culture. The remaining culture can cause anger, now listed as "We resent being ruled by a foreign culture" instead of "We yearn to join our motherland." 099

– Revolts can occur in any cities, not just those close to foreign borders, and culture of

dead civs can cause revolts. However, only border cities can flip. Cities can never flip from a master to its vassal. If a city can't flip, it loses one population on the third revolt and on subsequent revolts. 099c

– The occupation timer after conquest or revolt decreases only with a per-turn probability that depends on the same factors as the revolt probability, in particular the strength of the occupying force. The timer starts at no more than 3 turns.

Damage makes units less effective at suppressing revolts, and occupying units heal only as fast as in a friendly non-city tile.

Revolts can happen during occupation unless the city owner is at war with the owner of the foreign culture. 023

– "Cruel oppression" anger in a city makes revolts considerably harder to suppress. (Otherwise, Slavery would be a too obvious answer for anger from foreign culture.) 101

– The flipping-after-conquest game option is now called "No City Flipping after Conquest", i.e. its effect has been inverted. 101

– Tile culture is subject to a small exponential decay rate. The rate is increased for "stolen" tiles, i.e. tiles that no city can work but that could be worked if they flipped to a different owner. 099b

– The Creative trait grants 10 free culture upfront and only +1 culture per turn. Moreover, Colosseum is no longer sped up by that trait, and the happiness-from-culture abilities of Colosseum and Theater have been swapped. 908b

– K-Mod also makes changes to culture and revolts. AdvCiv keeps some of these – see the first few bullets in the next chapter –, and reverses others:

• Foreign culture strength no longer goes toward infinity when the current owner has close to 0% culture. 101

• Culture from trade routes disabled (still optional via XML). 125

• Reverted most of the K-Mod changes to building culture rates. 201
To balance out the remaining changes (especially to Free Speech), the threshold for Legendary city culture gets reduced on the low and medium difficulty levels. (On Monarch, the threshold begins to increase, matching the increasing tech costs relevant for Space victory.) 251

K-Mod

See also the K-Mod thread on CFC.

BUG (Better Unaltered Gameplay): Many BUG options are disabled by default in K-Mod and AdvCiv in favor of a lean interface that provides essential time savers but doesn't look too different from unmodified BtS. Please use the BUG menu (Ctrl+Alt+O) to adjust the settings to your own preferences. To import settings from another BUG installation, it should suffice to copy the respective `.ini` files to `My Games\Beyond the Sword\AdvCiv\Settings` – perhaps minus `Advanced Scoreboard.ini` because the column order string isn't fully portable between BUG, K-Mod and AdvCiv (cf. 085). Like all BUG-based mods, AdvCiv will restore its default settings if an `.ini` file or the whole `Settings` folder is removed.

BULL is included only partly in K-Mod. I've merged the Show Hidden Attitude Mod (change advc.sha), city bar help (change 186), added Sentry behavior to Fortify-Heal (004l) and

replaced BULL's pre-chop option with a more general pre-build command (011) that is triggered by holding down the Ctrl key while clicking on a Worker command. There are still several conveniences in BULL and other UI mods that are, so far, missing in AdvCiv; that said, AdvCiv also includes numerous usability tweaks beyond those in other mods (see 004 for a list).

For players who aren't familiar with K-Mod, I'm listing the major gameplay changes. For players who *are* familiar with K-Mod, I'm also listing the K-Mod changes that I've reverted (end of this chapter).

– Culture system: Culture levels (e.g. "Influential") matter less in culture wars, and specific city culture values matter more. Moreover, the range at which the culture of a city affects borders has been increased, meaning that the number of border cities and their culture are less decisive. The output of some culture sources has been reduced:

  • Free Speech increases culture by 50% instead of 100%.

  • The Sistine Chapel adds only 3 culture to religious buildings, not 5, and the added culture isn't doubled after 1000 years.

  • Culture from Great Work depends on the game era [progression adjusted in AdvCiv].

  • The Spread Culture mission only affects tile culture, i.e. it can no longer be exploited for an "espionage victory."

  • Regarding culture victory, note that tech costs in the late game have also been increased, i.e. both culture and space victory take longer than in BtS.

– City flipping: Revolts happen faster, but cities don't flip until the third revolt. "The net effect of these changes is that [...] it's a bit more predictable and less dependent on luck." (from the K-Mod changelog)

  The revolt probability is halved if a unit with the Leadership promotion is present.

– Vassal agreements: Can instruct vassals to start war preparations via "Let's discuss something else." The impact of vassals on number-of-cities maintenance is more noticeable than in BtS.

– The number-of-cities maintenance cost is no longer capped at 8 cities, meaning that very large empires can incur very high city maintenance. (See 570 for details.)

– Expendable units are chosen as defenders when no defender has favorable odds. (Lead From Behind mod component)

– When a religion spreads, one of the older religions is sometimes (randomly) removed.

– The AI never agrees to Cease Fire.

– Trades offered by the AI can come with a discount. The discount is foregone if the player makes a counter-proposal. (In AdvCiv, the offer can also include more gold than the AI would normally consider; change 026.)

– Espionage points accumulated by rivals against the player are treated as secret. Rival war weariness is only displayed when demographics are visible.

– The AI uses a greater variety of spy missions.

– Strengthened considerably:

- Serfdom +1 commerce on farms and plantations; -1 on towns
- All cargo ships +1 capacity
- XP from Great Warlord increased based on the number of units in the tile

– Weakened slightly:

- War Elephant +10 cost
- The Colossus +100 cost
- State Property doesn't remove colony maintenance

– Strengthened slightly (incomplete list):

- Vassalage -25% number-of-cities maintenance
- Mercantilism upkeep Low
- Grenadier +10% city attack
- Watermill +1 commerce initially but only another +1 from Electricity
- Drill I takes -15% collateral damage
- Protective trait boosts production of Security Bureau
- Aggressive trait boosts production of Jail; Jail gets -2 espionage (regardless of trait)
- Industrial Park +1 free Engineer (i.e. 2 in total), +50 cost
- Guided Missile +1 range
- Hippodrome grants 2 Artist slots (like Theater)
- Space Elevator another +50% production for spaceship parts
- Forest Preserve +1 commerce
- Environmentalism doesn't penalize corporations, yields +1 happiness from Public Transportation but has High upkeep. Environmentalism and Public Transportation both reduce bad health from population instead of granting good health.

Some that I couldn't put better, quoted from the K-Mod database page:

– "Barbarian Galleys get -10% strength"

– "Tech requirements for corporations changed: Civ Jewelers: Corporation, Sushi Co: Refrigeration, Cereal Mills: Medicine, Creative Constr.: Steel, Std. Ethanol: Combustion"

– "Mining Inc now uses Aluminium and Uranium instead of Gold and Silver (this should make Mining Inc slightly weaker, and Civ Jewelers more viable)"

– "Cereal Mills also has +0.25 food output" [also: Aluminum Co. +0.5 research]

– "In the diplomacy screen, pressing 'Lets stop this fighting...' will now bring up the trade screen with the AI's suggested peace terms. (Originally, it just made peace instantly without any trades – it was a horrible button.)"

– "Tech trades in which the receiving player is more than 2/3 of the way through researching the tech or when the tech is two eras behind the 'game era' [the recipient's era in AdvCiv; 550e] no longer count toward tech trade memory; i.e. they don't contribute to causing 'We fear you are becoming too advanced'."

– "Global warming has less of an impact on the map, but (in long games) causes unhappiness to those civs contributing to pollution. There's a new advisor screen [Environment tab on the Economics Advisor] about this."

Revised by AdvCiv:

– Changes made in order to smoothen or reinforce K-Mod balance changes:

  • Forest Preserve at Biology instead of Scientific Method. 901

  • Watermill takes 4 turns to build, and Lumbermill 5 (instead of 8 both). 902

  • Serfdom has Medium upkeep instead of Low. 912a

  • Pacifism costs 1 gold per military unit, not 0.5. 912b

  • Colossus is obsolete with Chemistry. 310

  • Great Lighthouse obsolete with Astronomy; cost 250 (300 in K-Mod, 200 in BtS). 310

  • Quechua has no bonus against Archers; cost is 15 (was 20 in K-Mod). 907b

  • Panzer 2 first strikes instead of 1.5; no free Flanking promotion. 907f

  • Ship of the Line strength 10 (as in K-Mod), +25% against Frigate (K-Mod 20, BtS 50), req. Iron (as in BtS), +1 move, increased bombard rate, cost -10. 905b

  • Ironclad +1 move (as in K-Mod), at Steam Power (no Steel required), +25% defense on Coast. 905b

  • Machine Gun +25% vs. Mounted units, not 10%. 909a

  • Meltdowns slightly more likely but, as in K-Mod, far less disastrous than in BtS. 652

– Changes kept from K-Mod versions prior to 1.45:

  • Scientific Method +1 research per specialist (1.45 moves this to Computers and has Scientific Method provide +10% commerce)

  • Lumbermill at Guilds with +1 commerce (902: regardless of river) but no production bonus until Replaceable Parts (1.45 gives them +1 production already at Guilds)

– Notable adjustments to K-Mod AI behavior:

  • AI razes fewer cities and less randomly; 116

  • uses Slavery and Drafting less aggressively, though still frequently; 121b, 017

  • uses malicious espionage only against civs it dislikes. 120

Reverted by AdvCiv:

– See Immortal Culture about reverted changes to culture and revolts.

– Production overflow is never invested into an additional unit as this had lead to balance problems with AI civs training multiple Archers per turn. Instead, excess overflow is converted into gold as in the Unofficial Patch, though with some tweaks to make this conversion more difficult to abuse. 064b

– Gifting Great People to the AI does not provide a relations bonus. 141

– No impact of global research on inflation; instead, adjusted tech costs based on difficulty for a more historically accurate tech pace, and Immortal and Deity games now

start on turn 10. 251

– Some minor balance changes undone that weren't pulling their weight in terms of added complexity (change 200):

- Eiffel Tower back at Radio

- No +2 production for Assembly Plant; no withdrawal chance for Musketeer; no free XP for Dun

- Drill does not lead to additional promotions – except Blitz (see next chapter)

- All corporations have the same maintenance multiplier (i.e. mostly as in BtS)

Important miscellaneous changes in AdvCiv

– Attacking a Worker or Settler no longer yields a captured Worker; the attacked unit just dies, i.e. no Worker stealing. 010

– Trade routes can only come from revealed cities, and the paths to those cities need to be revealed as well. Cities are temporarily unavailable for trade when in disorder. Trade along rivers doesn't require any tech (as in Warlords; BtS demands Sailing). When borders of a war enemy block trade, a naval blockade can override this. Connecting a resource on an island workable by a mainland city requires only a road, not a fort. 124

– When mountain peaks block off part of a landmass, the parts are treated as separate continents. This is relevant for the placement of Barbarians, colony maintenance and wonders that affect only one continent – though the main point is to make things easier for the AI. 030

– Map generation: Clusters of Gold, Gems made less common. Silver may occur on Grassland Forest. No Flood Plains on river corners. Jungle may occur on Plains. 129

– New algorithm for assigning starting locations. Should partition the available land more evenly than in BtS. Not enabled for all map scripts (list). 027, 108

– When a tile with unfinished Worker builds is left alone for more than 7 turns, the invested Worker turns begin to decay. This is mainly to disincentivize pre-chopping. 011

– Forest and Jungle provide only 25% defense, and none if the tile is owned by the attacker. 012

– Workers can no longer chop Forests and Jungles on unowned tiles. 119

– AI evacuates land units that can't defend well from cities that are about to be attacked and untenable. 139

– When the AI has more happiness or health than it needs, it may refuse to accept resources in trade or may trade away its only copy of a resource. 036

– Hereditary Rule grants 1 happiness per every 2 military units and +25% happiness from resources. 912c

– Financial trait grants +1 commerce only on tiles with a natural yield of at least 2 commerce (e.g. Coast) or a total yield of at least 3 commerce (e.g. riverside Hamlet). 908a

– Philosophical trait grants only +80% Great Person birth rate (not 100%). 908c

– Terrace culture rate reduced from 2 to 1. 908b

– Ikhanda and Rathaus effect on city maintenance reduced by 5 percentage points. exp.2

– Praetorian strength reduced from 8 to 7, cost from 45 to 40, starts with March, still no city attack bonus, and renamed to "Legionary." 907a

– Skirmisher loses 1 first strike; first-strike immunity moved from War Chariot to Immortal and Immortal has its anti-Archery bonus halved from 50% to 25%. 907d/e

– Fast Worker has only 2 moves but ignores terrain movement costs. 907c

– East Indiaman is a unique Frigate with 3 cargo spaces. Increased bombard rate of Frigate. 905b

– Galley, Trireme, Caravel, Frigate, Privateer +1 move. (Except Barbarian Galleys.) Cost of Galley and Trireme reduced from 50 to 45. Cost of Transport reduced from 125 to 100 and can't attack. Trireme has strength 3, but gets no bonus vs. Galley. 905(b)

– Attack Submarine and Submarine (now called "Nuclear Submarine") switch places in the tech tree. Both units get 6 first strikes, which should make them more dangerous for Battleship, but Destroyer gets to ignore first strikes. Battleship costs 250 instead of 225. Stealth Destroyer receives the same anti-air and anti-submarine abilities as Destroyer and 1 cargo space for missiles but also a cost increase from 220 to 270. 906

– Drill I enables the Blitz promotion for ships, but Blitz only allows one extra attack. 164

– Metal Casting tech cost reduced by 1/6, Optics increased by 10% to compensate. 306

– Divine Right cost reduced by 40%. 174

– The Spy unit can investigate rival cities, but (unlike in Warlords) with a mission cost. 103

– Research progress toward a tech makes that tech cheaper to steal. 120i

– No espionage slider until Alphabet. 120c

– Get to choose from more civics and religions when using the "Change civics/religion" Spy mission, or when negotiating peace or trading with a vassal. 132

– Courthouse allows two Spy specialists, Jail only one. 911a

– Grocer and Market (and Forum) cost 130 production instead of 150. 911b

– Aqueduct (and Baray, but not Hammam) costs 90 production instead of 100. 911c

– The Apostolic Palace grants only +1 production to religious buildings, costs 350 production (instead of 400) and allows 2 Priest specialists (instead of 0). The Shrines also allow 2 Priest specialists each (instead of 3). 179

  The Apostolic Palace can propose war against (non-full) voting members, but only full members are compelled to declare war. kekm.25

– Can no longer bypass the Theocracy restriction by gifting missionaries. 123a/ kekm.4

– Damage from nuclear weapons to units made less predictable; SDI interception chance reduced from 75% to 60% and cost increased from 1000 to 1500 production. 650

– Wonders that have been removed from the production queue generate no "fail gold."

Also no gold from national units. 123f

– On Emperor difficulty and above, some of the AI discounts have been reduced, but human civs face increased production costs, research costs, city growth thresholds and Great Person thresholds. The AI also starts with fewer freebies, in particular, without Agriculture on Immortal and Deity and without a second free Settler on Deity. Progressive AI bonuses are now based on the game turn instead of the AI era. 251/ 250e

– The default player count is 8 for the Standard map size. 137

– Granary stores only 40% food (except when playing with the new "No Slavery" option), but provides food also after starvation. 912d/ 160

– Worker moves are excluded from the options for showing foreign moves. To speed those options up further, the AI also patrols far less than in BtS. 102

– Switched the no-war-probability of Roosevelt and Darius, meaning that Darius can start wars when pleased, and Roosevelt – normally – cannot. 005a

– Suryavarman's favorite religion is Hinduism and his favorite civic Caste System. 005a

Start Points as Handicap (SPaH)

The SPaH game option gives Advanced Starts only to the AI civs. You might want to use this option if:

• You find the game too easy on moderate difficulty, and dislike the crass ongoing AI bonuses on the high difficulty levels, perhaps for reasons of immersion. A big head start for the AI can be taken to mean that the AI civilizations emerged earlier than yours, like how the Romans appeared later than the Egyptians, whereas the ongoing AI bonuses on Immortal difficulty are difficult to explain or overlook.

A big AI head start can narrow down the viable strategies in the early game though. It helps to assign start points unequally to the AI civs because this leaves some targets for early warfare, and a few far-ahead AI civs can't build all the early wonders.

• You find the game too easy on moderate difficulty, and dislike the big AI head start on the high difficulty levels, e.g. a Worker on Immortal. (Note that AdvCiv removes the free Settler on Deity.) Since Advanced Start replaces the AI freebies from the difficulty setting, you can reduce the AI head start by setting a modest amount of AI start points, e.g. Deity with 400 start points.

• You don't want every AI civ to receive the exact same freebies (e.g. because this puts AI civs that start have Hunting as one of their two starting techs at a disadvantage).

Usage:

Not entirely intuitive because, sadly, the Custom Game screen can't be freely modified. You'll need to check the "Advanced Start" option in order to unlock the "Base Points" box; without Advanced Start, the SPaH option has no effect. If you enter at most 9999 Base Points, then each AI civ simply receives the amount of points entered.

If you enter a five- or six-digit number, SPaH assigns start points unequally. The box is then assumed to contain two distinct numbers: a percentage in the two digits to the right and an integer in the three or four digits to the left. E.g. 80050 means 800 and 50%. The left number says how many start points the bottommost AI civ receives, i.e. the civ in the bottommost AI slot. The topmost AI civ (topmost AI slot) receives a *fraction* of the

bottommost's points equal to the percentage. In the example, that's 50% of 800 = 400. The remaining AI civs receive values in between those two bounds. (Combined with Monarch difficulty, the 800/50% setting should be a challenge closer to Emperor than to Immortal.)

SPaH never gives an Advanced Start to human civs, i.e. they start as normal with a Settler and Warrior or Scout. Humans receive starting locations based on the difficulty setting, i.e. the best ones on Settler and the (almost) worst ones on Deity. Then, if an unequal point distribution is used, the AI civs with the fewest points (top AI slots) are placed in the locations closest to the human civs, and the AI civs with the most points (bottom AI slots) are placed the farthest away.

Once the game has started, the start point distribution is shown on the Settings tab of the Victory screen. You'll also notice that the game doesn't start on turn 0. This is done to reinforce the notion that the human civs are late arrivals, and to have the game year match the overall development of the world.

An AI civ with a small number of points may have difficulties repelling a human Warrior rush, so I'd recommend against going far below 300. Unequal distributions are incompatible with team games, which is to say, they should technically work, but the assignment algorithm makes no effort to balance the teams' start points and locations. See 250b about how to enable Advanced Start and SPaH in scenarios.

General changes to Advanced Start (250c):

BtS allows start points to be converted 1:1 into production, which is usually far better than a 1:1 conversion into research. In order to make technologies a bit more attractive in Advanced Start, I've changed the conversion rate to 1:1.5 for production, meaning that cities, units and buildings now cost 50% more. Improvements, visibility and culture still seemed overcosted in comparison, so I've lowered the costs of these a bit.

Advanced Start costs are no longer adjusted to the game speed setting. This makes it unnecessary to manually adjust the start points entered on the Custom Game screen to match the game speed. When playing without SPaH and on a difficulty higher than Noble, the AI receives more start points than entered. This was apparently intended by the BtS developers but not correctly implemented. Conversely, human start points are no longer adjusted based on difficulty (this was working in BtS); humans get exactly as many points as entered.

For reference, the AI freebies for each difficulty setting, and the number of start points that would be needed in order to buy those freebies given the above changes to Advanced Start costs and assuming Standard map size (tech is more expensive on larger maps):

Difficulty	Prince	Monarch	Emperor	Immortal	Deity
Free initial items per AI civ	1 Settler 1 Warrior	1 Settler 1 Archer 1 Warrior Archery	1 Settler 2 Archers 1 Scout[see 250e] Archery Hunting	1 Settler 1 Worker 2 Archers[250e] 1 Scout[250e] Archery Hunting[250e]	1 Settler[250e] 1 Worker 3 Archers 2 Scout Archery Hunting Wheel[250e]
Worth in pts.	187	294	386	486	651

Includes 15 extra points for the 10 free initial production that the AI civs receive on all difficulty settings. In AdvCiv, the AI civs do not receive this free production after an Advanced Start.
On Monarch and below, if an AI civ starts with Hunting, the Warrior becomes a Scout.

When starting in a later era, the AI civs receive different free techs (cf.).

Rise and Fall (R&F)

If the R&F game option is enabled on the Custom Game screen, the game gets divided into several *chapters* (of history). During the first chapter, you play as the leader and civ set on the Custom Game screen (random also works). Once a chapter ends, you're asked to select one of the other civs in the game. During the next chapter, you control the selected civ, while your previous civ is controlled by the AI. After each chapter, a score is awarded that measures how much your civ has improved since the start of the chapter. If you manage to fulfill a victory condition, the game ends, and you score a premium, which is, again, based on the standing of your civ at the start of the chapter. To maximize the total score, you need to pick civs that are doing badly but have the potential for improvement. If you play well, civs will "rise" under you control, and begin to decline after AI control resumes – this is the "fall" part. That is, unlike in the Rhye's and Fall mod, there is no special collapse mechanism. There is no connection with the "Rise & Fall" expansion for Civ 6. The High to Low challenge in Kael's Assimilation mod has been an inspiration.

Supported and recommended game settings: Single-player only, no teams or Permanent Alliances. The number of chapters is 6 on Quick speed, 7 on Normal, 8 on Epic and 9 on Marathon, provided that there are enough civs in the game. Since a different civ is supposed to be played in each chapter and some are usually eliminated before the final chapter, it's advisable to configure at least 1.5 times as many civs as chapters. If you dislike playing on Huge maps, you can fit enough civs on a Large map (up to 16 I'd say) or even a Normal map (up to 12) if you set the sea level to Low (assuming a standard map script like Fractal or Continents). You may want to take a look at the "Score" tab of the Victory screen at game start in order to verify that the number and length of chapters is as intended.

I haven't tested unusual settings much, but R&F should, in principle, work with any map script and any scenario where the civs start without cities. Advanced Start should be fine too, but not SPaH. Starting in a later era than Ancient results in fewer chapters.

R&F is intended to be played at a difficulty that makes it challenging to take a civ from a rank near the bottom to a rank near the top over the course of a chapter. That's going to be about three difficulty levels lower than what would normally pose a challenge. When it comes to expenses for units and cities (of human and AI civs alike), R&F will apply difficulty levels one higher than those configured on the Custom Game screen (i.e. Prince for the AI civs); see 708 for details. This is done because expenses can feel quite insignificant when playing several levels below one's usual difficulty level.

Objective: The objective is to maximize what I'm calling the *Rise* score. When the game ends, that score is computed as the sum of the *chapter* scores plus a premium for victory. What the scoreboard on the main interface shows are scores based on population, technology etc.; this score is computed as in BtS, and I'm calling it *CIV* score to

The CIV score from the scoreboard on the lower right feeds into the chapter score on the left of the Score tab, which feeds into the Rise score on the right of the Score tab.

distinguish it from the chapter score. Chapter score is based on the change(!) in CIV score and rank since the beginning of a chapter. Once the game ends, the final rank between "Dan Quayle" and "Augustus Caesar" is computed from the Rise score. (In non-R&F games, the rank is computed from the CIV score and game end turn.)

Delayed scoring, AI changes: A chapter isn't scored right after it ends but during the subsequent chapter. This is done to disincentivize shortsighted plays toward the end of a chapter, e.g. conquering cities that aren't tenable in the medium term. The first chapter is scored in the middle of the second chapter. With each chapter, the scoring delay decreases a little; the penultimate chapter is scored one quarter into the final chapter. The exact turns on which scoring happens are shown on the upper half of the Score tab. On those turns, a popup with a breakdown of the chapter score appears.

A side-effect of delayed scoring is that you have a stake in two civs during the first portion of each chapter: the one you're currently playing and the one from the previous chapter. This creates an awkward incentive for helping the civ from the previous chapter. Similarly, toward the end of a chapter, there can be an incentive for helping whichever civ you're going to take control of in the next chapter. In order to limit such collusive help, I've made the following changes to the AI:

– Doesn't accept unsolicited gifts, and asks for gifts less frequently than normal. No units can be gifted.

– Rejects offers that are heavily lopsided in favor of the AI. This can also mean that the AI refuses to make peace in exchange for a valuable technology.

– Is reluctant to get paid for things of questionable value to the player: civics and religion changes, embargoes, sponsored war and brokered peace.

These restrictions don't apply: to civs that have already been scored; to vassals of the player; nor to any civs that outrank the player on the scoreboard. During the final chapter, the restrictions only apply to the civ from the penultimate chapter (until that civ is scored).

Intermediate periods: In between every two consecutive chapters, there is an "inter-mediate" period of 5 to 10 turns (depending on game speed) during which you control no civ. During an intermediate period, most elements of the user interface aren't updated, meaning that you have practically no vision and agency. A countdown on the upper right shows how many turns remain until the start of the next chapter. The main purpose of the intermediate periods is to hinder military cooperation between the civs played in two consecutive chapters.

Civ selection: When an intermediate period ends, you're prompted to select a civ for the upcoming chapter. Each civ should be played for at most one chapter. Sometimes, this is impossible because too many civs have been eliminated already; therefore, it's allowed to play the same civ repeatedly, but the chapter score is halved then. Playing the same civ for two chapters in a row is not possible, and any vassals of the most recently played civ are off-limits too.

The selection popup lists civs in order of

Popup for civ selection

recommendation: First, non-vassals not previously played, inversely ordered by rank, then, vassals not previously played and, last, civs previously played. Civs known to any previously played civ are listed by name, the others as "unknown." Hovering over an entry shows the civ's current CIV score, whether it's a vassal, and if so, if capitulated or voluntary and the identity of the master, and lists any war enemies. A victory stage is shown if the civ is close (at least stage 3 out of 4) to fulfilling a victory condition. All this information is provided even if the name of the civ is unknown. It's often unwise to take over a civ that is at war with a stronger opponent. If you select a voluntary vassal, you'll be able to cancel the vassal agreement at any time. As a capitulated vassal, you'll have to acquire enough land and population in order to break free.

Chapter scoring function: The computation of the chapter score isn't entirely simple, but it may suffice to know that the more the CIV score has increased between the start of the chapter and scoring, the higher the chapter score.

• When a chapter is scored, 0 to 100 points are awarded based on the rank *r* of the scored civ on the CIV scoreboard at the time of scoring:

Chapter score breakdown (Score tab)

$$100 * (\text{worst rank} - r) / (\text{worst rank} - 1)$$

That's 100 when ranked first (*r*=1) and 0 when ranked last (worst). In this context, the civ currently controlled is always considered to be ranked behind the civ that is being scored; this way, success in the current chapter can't hurt the score for the previous one. A score for the *initial* rank, i.e. the rank at the time that the scored civ came under human control, is computed in the same way, and subtracted from the score for the current rank. Thus, the total score from rank is maximized (100 points) by starting in the last place and climbing all the way to first. A civ that starts in rank 1 can at best gain 0 points from rank by staying in rank 1, and otherwise loses points.

• Points are also awarded directly for CIV score; otherwise, it would be pointless to continue a chapter when it's clear that the current civ can't climb another rank. The CIV score of the next best rank is used as a point of reference. In the example on the right, Rome is currently fourth, so its score of 929 is divided by that of the fifth rank (Ethiopia, 922), and the resulting percentage (101) is added to the chapter score. At the start of the chapter, Rome's score of 352 was 65% of the fifth rank (Charlemagne, 541), and that percentage is subtracted, resulting in a net gain of 36 points from CIV score.

Example: Current scoreboard (left) and scoreboard at the start of the current chapter

• If the chapter was ended prematurely through retirement (see "End of chapter" below), the chapter score from rank and CIV score is increased by half the percentage of the turns remaining at the time of retirement. For example, if 14 out of 65 turns remain, the portion of unspent time is 22%, so the chapter score is increased by 11%.

If the bottom line is -10 or worse, a square root is applied in order to reduce the loss of points. This is done so that a single failed chapter can't ruin the total Rise score.

End of chapter: All chapters have the same length except for the last one, which can be a

bit longer or shorter than the others. The chapter length depends on the number of chapters and the game speed. To see how many turns remain, hover the mouse over your civ's flag on the main interface. During the last three turns of a chapter, a countdown is shown directly on the main interface, and, at the beginning of the final turn, there is a popup reminder.

As the end of a chapter approaches, you may want to try putting your civ in a state that its AI leader can work with because the civ will be under AI control for some time before scoring happens. E.g. Gandhi might not continue your war of conquest. Don't rely on pre-programming decisions for the AI: When the AI takes over, the mod clears your slider positions, governor settings and the queues for city production, research and unit missions.

You can end a chapter early by selecting "Retire" from the main menu. This button, which normally concedes the entire game, works very differently with R&F. By retiring early, you earn a bonus to your chapter score (see "Chapter scoring function" above) and the chapter continues on AI Auto Play until its regular end turn. One situation in which retiring makes sense is when your civ is getting so far ahead of the others that the AI might run away with the game once the chapter ends.

Chapters				
1	Zululand	Turn 0 to 64 (1440 BC)	**57 points**	**Scored on Turn 118**
2	England	Turn 73 to 137 (540 AD)	**107 points**	**Scored on Turn 191**
3	Rome	Turn 146 to 210 (1500 AD)	154 points	Scored on Turn 264
4	**Aztec**	**Turn 219 to 283** (1826 AD)	31 points (**turns remaining: 34**)	Scored on Turn 337
5	?	Turn 292 to 356 (1936 AD)	?	Scored on Turn 410
6	?	Turn 365 to 429 (2009 AD)	?	Scored on Turn 483
7	?	Turn 438 to 499 (2049 AD)	?	Scored on Turn 499
		Total so far:	349	

List of chapters on the Score tab, showing start and end turn (with end year), chapter score and scoring turn for past chapters, the current chapter and future chapters. The chapters with score values in bold have already been scored; score values not shown in bold indicate the score that would be awarded if the chapter were scored right now.

Game end: The game ends immediately when one of the BtS victory conditions is fulfilled by any civ. If the previous chapter hasn't been scored yet, it is scored now. If an AI civ wins, then the current chapter is scored as well and the Rise score is simply the sum of the chapter scores. If the human civ wins, then the current chapter is not scored, i.e. rank and CIV score don't matter. Instead, a victory premium is added to the total score of the earlier chapters. This is, once again, a bit complicated; in short, the incentive is to win as early as possible if it can be done from a poor position, but not to win a quick victory by picking a civ that is already about to win.

- Premium **P:** 150 points plus 150 for each chapter that hasn't started yet.

- The premium above is reduced based on the rank r_0 at the start of the current chapter (initial rank) by subtracting

    (**P**/1.5) * (worst rank – r_0) / (worst rank – 1).

    However, even when the initial rank is 1, the premium is never reduced below a lower bound of **P**/3.

Rise Score at game end	
+164	from sum of scored chapters
+154	from finished but unscored chapters
+600	from victory, compensating for 3 missed chapters
-36	from initial rank (11. of 12)
+51	for 51% time unspent in current chapter
+233	(+25%) from Prince difficulty
1166	in total

Rise score breakdown (Score tab)

- In the endgame, score isn't a good measure of a civ's standing. Therefore, whenever R&F considers ranks, those civs close to a victory condition (stage 3 or 4) are con-

sidered to be ranked higher than those not close to any victory, and civs at stage 4 outrank those at stage 3.

- Regardless of the initial rank, the percentage of turns remaining in the chapter in which victory has been achieved is added.

- Finally, the Rise score is multiplied by a factor based on the difficulty setting.

On the Hall of Fame screen, the Rise score is shown in the "final score" column. Use the "score victory" filter in order to display only results of R&F games.

A defeat of the human civ does not cause the game to end unless it happens in the final chapter. Instead, human defeat causes the game to continue on Auto Play until the chapter ends, and then a new civ is selected as normal. A defeated civ is scored immediately and is treated as having 0 CIV points.

PerfectMongoose (PM)

The PM map script is based cephalo's PerfectWorld3 (PW3), which employs models of plate tectonics, wind patterns and hydrology. The basic ideas are best described in the old CFC thread for PerfectWorld2 (PW2). LunarMongoose ported PW2's successor PW3 from Civ 5 to Civ 4, incorporated changes by AIAndy and Fuyu, and made changes of his own, which are listed in the PM thread on CFC.

- PM vs. Tectonics: LDiCesare's Tectonics script, which is included in BtS, also models plate tectonics and wind but, as far as I can tell, in simpler ways. PM has about three times as many lines of code as Tectonics does if that's any indication. That said, Tectonics offers some unique options like "Mediterranean". An updated version of Tectonics is also included with AdvCiv (change id 021a).

- PM vs. Totestra: Totestra is a PW2 fork by vktj. Totestra offers more custom map options than PM, but PM has a more sophisticated noise generator. (Totestra should be compatible with AdvCiv but isn't included in the mod.)

I've customized PM (change 021b) because I felt that the starting locations were too unbalanced and it played too differently from the standard map scripts. In particular, there was little arable land.

In many ways, PM can supersede the Fractal map script, but, since PM tends to generate more continents and more chokepoints than Fractal, the space available for expansion tends to be distributed less evenly. Other caveats:

- Huge maps might take a couple of minutes to compute.

- The "Old World Start" option sometimes only manages to reserve a small New World.

The "Plate Tectonics" option tends to produce somewhat more massive continents and mountain ranges. It uses the older landmass and elevation generator from PerfectWorld 2.

Mixed Continents

A variant of jam3's Continents and Islands, which, in turn, is based on the BtS script "Big and Small" by Sirian. I've called my version "Mixed Continents" because it doesn't necessarily involve islands – by default, the two landmass types are normal continents and small ("snaky") continents. It's possible to place the two landmass types in two separate regions of the map, but, by default, they're mixed together. (For separate regions, K-Mod's

"Not Too Big or Small" script may be the better choice.) I've disabled jam3's terrain options because I found them a bit arbitrary, and, instead of the "Add Water" option, the sea level setting is now used. ("Big and Small" ignores the sea level.) I've also added a world-wrap option and tweaked the land/ sea ratio and map dimensions so that the same player counts can be used as on e.g. Fractal. For some more details, see change id mxc.

All-AI games

An all-AI game can be a quick way to to get an impression of the AI behavior in AdvCiv or to check how well the mod handles certain game settings. The procedure is as follows:

1. Make sure that cheats are enabled (`CheatCode = chipotle` in `My Games\Beyond the Sword\CivilizationIV.ini`). This is a prerequisite for Debug mode and AI Auto Play.

2. On turn 0, press Ctrl + Z to switch into Debug mode. This makes the entire map visible.

3. Unless the game is set to Noble or Prince difficulty, if you want your own civ to have the same starting conditions as the other AI civs, you'll have to use the WorldBuilder to give additional techs and units to your civ (if the difficulty is higher than Prince; see the table above the Rise & Fall chapter), or to all other civs (if the difficulty is lower than Noble).

4. Press Ctrl + Shift + X to have the AI take control of your civ. A popup lets you choose the number of turns to be spent on AI Auto Play. (A mnemonic for the shortcut: control is shifted to the AI for x turns.) AI Auto Play can also be interrupted through Ctrl + Shift + X, though you may have to press these keys repeatedly because key presses are not received reliably during AI Auto Play.

AdvCiv makes a few improvements to the AI Auto Play mod component (change id 127):

• The player's civ plays by the same rules as the AI civs; in particular it plays on the same difficulty level.

• While both AI Auto Play and Debug mode are enabled, messages about major game events such as declarations of war and conquered cities are shown as if the player's civ had perfect knowledge of the game state.

• Unless interrupted, AI Auto Play ends at the start of a human turn. This means that no diplo popups can occur on that turn. (That said, it can still happen that the player is prompted to vote on a UN resolution right after AI Auto Play ends.)

Performance

Although the UWAI component is computationally expensive in large games, as of version 0.99, the mod appears to be faster than K-Mod even when the number of civs is increased beyond 18. In a test with 31 civs (i.e. $32=2^8$ players when counting the Barbarians; this number has some computational advantages) on a map with 160 x 100 tiles, the first 300 turns on AI Auto Play (see All-AI games) took fewer than 30 minutes with AdvCiv and more than 50 minutes with K-Mod. K-Mod, in turn, is significantly faster than BtS. Note that changes to the civ limit (normally 18) require the DLL to be recompiled; see under For developers. For benchmarking, the shortcut Ctrl+Shift+B can be used (cf. BM in the appendix). The performance gains in AdvCiv have been achieved through various low-level optimizations across the Game Core DLL (cf. advc.opt).

Known limitations

– PitBoss and PBEM (play by e-mail) mode haven't been tested.

– Some of the Custom Game settings have been tested only superficially through AI Auto Play, and of course I haven't been able to test every combination of settings.

– If the mod folder "AdvCiv" is renamed, the name also has to be changed in `Assets\ XML\Art\CIV4ArtDefines_Misc.xml` and (twice) in `Resource\Civ4.thm`. Otherwise, the mod will crash while loading: "Failed to initialize the primary control theme." This is a side effect of change 002b (enlarged fonts). That change also makes it impossible to install AdvCiv into `My Games\Beyond the Sword\Mods`.

– The current version of the mod should be able to load savegames from all earlier versions but no savegames from other mods nor from unmodified BtS.

– The mod has only been tested with the international (MULTI5/ EFIGS) version of BtS. German translations are almost complete, though hardly tested. I don't recommend using one of the Romance languages as I've only translated parts of the new game text.

– One player has reported a potential issue with AdvCiv and Windows 7 user account control, so I've listed some steps that might help here. The dated but detailed instructions here could also be helpful. (Addendum: If a compatibility mode is needed, some say that it's best to use Vista SP2, others say Win 7.)

– The Work in progress section at the end of this manual lists non-critical open issues. (Most of them are not really work in progress anymore because v1.0 is supposed to be the last major release of this mod.)

For developers

The modified Game Core source code files are available on GitHub. In all modified files, changes and additions are labeled in-line with "`advc.id`" or enclosed in XML style: `<advc.id> ... </advc.id>`, where `id` is one of several three-digit numbers that I've (pretty arbitrarily) assigned to sets of related changes. Lately, I've started to replace some of the numerical ids with letters, e.g. "advc.opt" for miscellaneous performance optimizations. Nonfunctional changes (cf. 003) are marked with just "`advc`" (no id number).

Changes from K-Mod-Extended are instead marked with `kmodx` (these are also in K-Mod 1.45, but I had merged them into AdvCiv before 1.45 was released), those adopted from Kek-Mod with `kekm`. and contributions by Erik (devolution on CFC) with `Erik`. AI changes from the LoR SDK ModMod are tagged with `cdtw`. Nightinggale's GameFont Display is tagged with `gfd`.

I've gathered bugfixes and tweaks that other mods could easily adopt *from* AdvCiv in this Git branch based on K-Mod 1.46. I've committed DLLs, so the branch is playable too.

UWAI is documented mostly through comments in code (more verbose documentation exists but is out of date); the appendix (id 104) only describes the integration of UWAI into BtS. Parts of UWAI are adjustable through `AI_Variables_GlobalDefines.xml`. If you've enabled logging (`LoggingEnabled=1` in `My Games\Beyond The Sword\CivilizationIV.ini`), you can change `REPORT_INTERVAL` in `GlobalDefines_devel.xml` in order to have UWAI write AI internals to `My Games\Beyond The Sword\Logs\uwai.log`. The logs are formatted in Textile. I've been using Borgar Þorsteinsson's Textile-JS to parse it (free web interface). In

Debug mode (Ctrl + Z), AI war plans can be checked in-game by holding down the `Alt` key and hovering over the AI leaders on the scoreboard; same as in the BBAI mod. If `UWAI_SPECTATOR_ENABLED` is set to 1 in `GlobalDefines_devel.xml` and Debug mode is enabled, then announcements about AI war preparations are shown during AI Auto Play. Debug mode, WorldBuilder and AI Auto Play can be enabled for network games through XML; see 135c for details. See 007 for a crib sheet about all available logfiles.

Unlike most mods, AdvCiv sets the maximal number of civs (`MAX_CIV_PLAYERS`) not in `CvDefines.h` but in `CvEnums.h`. The standard version of the mod uses `MAX_CIV_PLAYERS=18` (like BtS) because unused player slots negatively affect performance. Due to various code optimizations, that overhead should be almost insignificant at this point, e.g. less than 10% (perhaps less than 5%) when increasing `MAX_CIV_PLAYERS` to 31 while using only 18 civs. Still, increasing `MAX_CIV_PLAYERS` in AdvCiv would break savegame compatibility with earlier versions of the mod. For mods derived from AdvCiv, going to 31 (i.e. 32 players when counting the Barbarians, a computationally advantageous number) would seem like a good idea, seeing that 18 aren't really enough to fill a Huge map at Low sea level and that some spare civs are desirable for colonial vassals.

Rebuilding CvGameCoreDLL.dll: If you've done this before for other mods, then it's just a matter of setting the usual file paths in `CvGameCoreDLL\Project\Makefile.settings`. Otherwise, see step 5 in this old guide by Asaf about the paths; `CIV4_PATH` corresponds to `CIVINSTALL`. As for installing the Windows SDK and Visual C++ toolkit, see step 3 under "Setup from Scratch" in Leoreth's guide. I use slightly different versions (see appendix), but those in the guide should also be fine. Step 2 describes how to install Visual Studio (VS) 2010 Express, which is what I've been using; any newer version of VS can be used as well, though it appears that VS 2010 still has to be installed. Make sure to select VS 2010 for C++ in the installer. To build the DLL, open `CvGameCoreDLL\Project\AdvCiv.sln` in VS and select "Build" from the "Debug" menu (or press F7). The build configuration can be changed in the "Configuration Manager". Nightinggale's guide contains some information about the available configurations. See "advc.make" in the appendix below about configurations added by AdvCiv.

To attach the VS debugger, I recommend creating a copy of `My Games\Beyond the Sword\ CivilizationIV.ini`, e.g. named `AdvCivDebug.ini`. It's easiest to place it in the same directory as `Civ4BeyondSword.exe` (`CIV4_PATH` directory). In the copy, set `FullScreen = 0` and `Mod = Mods\AdvCiv`. You may also want to reduce `ScreenHeight` a little (rationale) and make some other debug-friendly settings: `CheatCode = chipotle`, `LoggingEnabled = 1`, `AutoSaveInterval = 1`. Then select the "Debug" configuration in VS 2010 and under "Configuration Properties" - "Debugging," enter `Civ4BeyondSword.exe` with its full path(!) into the "Command" field and `ini=AdvCivDebug.ini` into "Command Arguments". (VS will write that information into `CvGameCoreDLL\Project\AdvCiv.vcxproj.user`; that file can also be edited directly.) You can then "Start Debugging" via the "Debug" menu.

Steam users need to install Steamless for debugging.

A note about the **GNU Public License**: A copy of that license is included in the BUG Doc folder only for the sake of the AI Auto Play mod. AdvCiv as a whole is not (at this time) distributed under that license and I don't believe the BUG mod is, either.

Appendix

Most of the change ids are assigned pretty arbitrarily. Changes since version 0.8 are also tracked through Git: commit history.

001	Minor bugfixes (not a complete list)
See also	Fixes merged from other mods: kmodx, kekm. mnai. 104z fixes a bug in `CvPlayerAI::AI_eventValue`.

Corrected the city culture output shown during disorder by K-Mod/BUG with enabled Building-Actual-Effects option.
AI refuse-to-talk duration was lowered purely based on war success ratio in BtS, not absolute war success. This way, killing a single stray unit before stack combat could make the AI willing to talk. Added a lower bound for enemy war successes.
Barbarians no longer spare a single target city per continent from pillaging. In BtS, they apparently try to conquer that city intact. (`CvUnitAI::AI_pillageRange`) And Barbarians don't target cities protected by the Great Wall. (`CvPlayerAI::AI_findTargetCity`) Obsolete; Barbarians don't have a target city at all now through change 300.
Only cities whose location the AI knows or is able to deduce can be per-continent target cities. The corresponding visibility checks were missing in various places, but mainly in `CvPlayerAI::AI_findTargetCity`. Through `CvPlayer::handleDiploEvent`: An AI vassal will now target no city in particular when asked by its human master to attack a city that is not revealed to the vassal. (Don't want to filter those cities out in the menu because a player isn't necessarily supposed to know which cities are revealed to its vassals.)
Reduced the chance for building a "priority fort" from 80% to 20%, which, I suspect, was intended. (No clue if it's wise.)
Since K-Mod 1.44, Gunship had been able to capture cities if they contained a visible non-combat unit. Reported by Zholef here. Also fixed in K-Mod 1.45, but I've kept my own fix instead.
Eliminated AI civs can no longer do diplomacy. In BtS, there is a slim chance for this to happen right after an AI is eliminated by another AI.
AI no longer trades for resources needed only for obsolete units (merged from Better BUG AI).
Integer overflow in K-Mod code when a large amount of culture is added through WorldBuilder. Thanks to xaster for pointing out the issue on CFC. Also fixed in K-Mod 1.46.
BBAI attitude cache updated when the ranks on the scoreboard are updated.
The dot map overlay shows up correctly after loading. Bugfix by EmperorFool: source *Tbd.*: Overlay isn't cleared when regenerating the map. And stays disabled after AI Auto Play. And doesn't store the most recently selected color in savegames. And appears to be cleared when Python scripts are reloaded (if this doesn't crash BUG entirely; see 009b).
When enemy units and neutral units are together in a tile, the unit model shown on the map is guaranteed to be an enemy unit. (K-Mod bug in `CvPlot::updateCenterUnit`)
A resolution for mutual defensive pacts can't be signed if all non-vassal members already have signed defensive pacts. (`CvGame::isValidVoteSelection`)
In `CvFractal::tectonicAction`, a variable was unused, and this looked like a bug. I'm not sure though; the maps look as before. It seems that only the Continents script calls that function. (I'll probably also use it for the "True Starts" script; see tsl.)

Two bugs fixed in the `calculateTradeRoutes` function (`TradeUtil.py`, `CvExoticForeign Advisor.py`). I don't think either of these had any impact in AdvCiv, but they were still bugs. Credits: Leoreth (Dawn of Civilization mod) Git commit 1 2

`autologEventManager.py` (Autlog mod) had assumed the active player slot to be 0 in a few places. There might be other issues with scenarios (which don't usually give slot 0 to the active player) – I haven't tested it; these are just errors I came across when browsing through the code.

K-Mod shows cities as Barbarian if the true owner hasn't been met (city revealed through map trade). Now showing the true color. In `CvCity::getCityBillboardSizeIconColors`.

If the AI starts the game with a free Worker, then the starting Settler is moved before the Worker. (In K-Mod, the first Worker turn is always wasted.) Implemented through `CvPlayerAI:: AI_movementPriority` and `CvUnitAI::AI_handleStranded`.

When a unit that is selected by the active player dies (in combat or disbanded), the per-turn income shown on the upper left is immediately updated to reflect possible changes in expenses (unit cost and supply). In `CvUnit::kill`.

Corrected the id of K-Mod's "Simple Unit Selection" option in the DLL (`CvGame::selectUnit`); the option had no effect previously.

From Mongoose Mod changelog
12-14 Dec 2012: "*bug in CvUnitAI::AI_betterPlotBuild [...] it was thinking every plot bridged two plotgroups*"
15 Feb 2013: "*vanilla bug with the AI evaluation of […] worldsize-scaling -1 quantity value on Broadway, Rock 'N Roll, and Hollywood's free resources*" (in `CvCityAI::AI_buildingValue`)

On the city screen, use the art style of the city owner even if another player is inspecting the city. Based on this bugfix by Leoreth (Dawn of Civilization mod). I've fixed it entirely through `CvPlayer::getUnitArtInfo`.

Set "time played" back to 0 when regenerating the map. (`CvGame::regenerateMap`)

`canParadrop` and `canParadropAt` had not been correctly exposed to Python (*Credit*: Roamty; link)

In `CvCityAI::AI_getPlotMagicValue`, the `YieldWithBuild` value needs to be taken times 100 to match the scale of `AI_getYieldMagicValue`.

This bug was introduced with the BtS expansion and probably mattered only for the `iPopToGrow` computation in `CvCityAI::AI_yieldValue`. Even there, the impact seems to have been minor because happiness is normally the dominant limiting factor for (planned) AI population growth. *See also*: The bugfix is important for 113.

Fixed a potential issue with the BtS limits on withdrawal, evasion and interception chance in `CvUnit::isPromotionValid`: Once a unit exceeded a limit, it was unable to receive any promotion. That would be a problem when a mod increases the respective chance value in `Civ4UnitInfo.xml`. Now only promotions that increase the chance value are blocked.

The `pyAssert` function in `CvUtil.py` now actually raises an assertion error. Somewhat important to let `findInfoTypeNum` (same module) tolerate empty strings then; will otherwise render scenarios unplayable that cause no problems in BtS, specifically the Accurate Earth Maps by Laskaris.
Credits: More Naval AI (lfgr), CFC user wfeiger for reporting the problem with Laskaris's maps (CFC post).
(Also adopted – not really a bugfix: Git commit)

The AI doesn't cheat with visibility when selecting a target city for a Trade mission (Great Merchant) in `CvUnitAI::AI_tradeMissionValue`.
Credits: More Naval AI (Tholal)

Wrong type of AI strategy bitmask in `AI_getImprovementValue`,

`AI_updateSpecialYieldMultiplier` (both CvCityAI, having to do with AI yield adjustments). Logical operators used with bitmask in `CvPlayerAI::CvPlayerAI::AI_espionageVal`; something about the counterespionage mission.
All three in K-Mod code.
See also: Found after turning the bitmasks into enumerators ([advc.enum](#)).

`CvPlayerAI::AI_unitValue`: Wrong parameter for the `isCarrierUnitAIType` call (BtS bug). This probably meant that the AI was able to recognize special cargo units only through the AI types stated in C`iv4UnitInfos.xml`. So only a problem for mod-mods, maybe.

In the BBAI code for (automated) air recon, a visibility check had been flipped in `CvUnitAI::AI_exploreAirPlotValue`, and, in `CvUnitAI::AI_exploreAir2` (renamed by AdvCiv to `AI_exploreAirRange`), the tiles around the airbase had been evaluated instead of the tiles around the candidate recon tile. On the bottom line, the recon target tile was chosen randomly with a bias only for maximal distance from the airbase.
See also: [650](#) uses that function also for AI air recon. [029](#) makes some AI tweaks.

In `CvUnitAI::AI_handleStranded`, units had been moving toward any coastal tiles including lakeshores. Now only seashores are targeted.

`CvPlayerAI::AI_doDiplo` failed to check for a trade connection before demanding resources from a vassal.

Fixed a perhaps inconsequential copy-paste error in `CvDeal::isUncancelableVassalDeal`.

K-Mod's `ROUND_DIVIDE` function was incorrect for fractions between 0 and 1. (`CvGameCoreUtils.h`)

The Partisans event had used tile culture to determine the previous city owner's culture level instead of city culture. Also (not a bugfix I suppose), I'm subtracting 1 from the culture level so that e.g. cities with "poor" culture don't spawn any partisan units.
Credits: Reported [on CFC](#) by SmokeyTheBear – who also suggested subtracting 1 and found another bug in the Partisans event that AdvCiv fixes through [003y](#).

Flipped sign in the AI evaluation of specialist experience (`CvCityAI::AI_jobChangeValue`). That ability is unused for (non-super) specialists, so this only matters for mod-mods (possibly).

Clear popups of non-human players before saving in `CvPlayer::write`.
Tbd.: Find out under which circumstances the EXE adds popups to non-humans. (Update a couple of years later: Looks like the EXE no longer does that. Probably a problem I had introduced and fixed without noticing.)

When a city gets destroyed and replaced due to a change in ownership (`CvPlayer::acquireCity`), don't check if the city tile is valid for local units until the new city is in place. Because the tile may flip to a third party for an instant.

A K-Mod 1.45 change in `CvCity::getProductionDifference` had (accidentally) caused angry citizens to consume food even during the production of a Settler or Worker. I've also added a loading screen hint about the underlying Vanilla/BtS rule.
Credits: Bug reported by CFC user carp.
See also: K-Mod [Git commit](#) introducing the bug. CFC [thread](#) where a Firaxis developer explains why angry citizens don't consume food. [Post](#) by karadoc, half a year before the Git commit, explaining that he's aware of the BtS rule and wants to keep it (reluctantly).
Tbd.: Maybe treat the population as 1 less while producing a settler or worker. That would seem more consistent. Still doesn't exactly make sense. (It would if workers and settlers consumed population as in Civ 3.)

K-Mod 1.44+ (through this [Git commit](#)) had chosen the "Big Espionage" strategy (`CvPlayerAI::AI_updateStrategyHash`) based on the AI espionage commerce weight and the AI espionage weight (in part) based on "Big Espionage", creating a feedback loop. The problem was

probably introduced when `CvPlayerAI::m_iEspionageWeight` was repurposed as a cache for the (final) espionage commerce weight.
Credits: CFC user SuperXANA made me aware that the AI uses "Big Espionage" very often.
See also: Might fix this issue raised in the "Dawn of the Overlord" thread.
Tbd.: Similar problem with "Espionage Economy" and `CvPlayerAI::AI_updateCommerce`; not sure if it has adverse consequences.

In `AI_techUnitValue`, BtS/BBAI/K-Mod code for incentivizing oceangoing ships had had no effect.

`CvGame::onGraphicsInitialized`: When loading a savegame in which the active player owns no units, the camera had centered on some (seemingly?) arbitrary tile, not necessarily a revealed tile. Now it centers on the active player's starting location in that case. *See also*: Based on a workaround (004j) that moves the camera after regenerating the map.

When a city was selected without opening the city screen, `CvGame::shouldDisplayUnitModel` had shown a 3D model next to the production queue only if a unit was selected before selecting the city. Now the model gets shown regardless of selected units.

The unused `AdvancedStartCostIncrease` XML tags for improvements and routes had increased the cost based on improvements or routes owned by any player – only those placed by the current player should count.

For random events involving two civs, it looks like `CvPlayer::applyEvent` hadn't properly checked whether both civs have been met before notifying the human player.

Looks like BtS and the unofficial patch didn't take the Random personalities option into account in `CvGame::addPlayer` and `CvPlayer::getSplitEmpireLeaders`. I haven't tested it, but, hopefully, colonial vassals will now receive a random AI personality that (usually) differs from the leader appearance.

Added code to `CvPlot::changeVisibilityCount` that works around a problem with the interaction of nuke visibility and Fallout replacing a sight-blocking feature (Forest or Jungle). *Tbd.*: A proper bugfix.

The price charged by the AI for an embargo (`CvPlayerAI::AI_stopTradingTradeVal`) had been counting all deals of the AI, not just those with the embargo target. (bug in vanilla Civ 4)

"No Action Recommendations" player option had been ignored at game start. Fixed by setting the dirty-bit for colored plots in `CvPlayer::setOption`.
See also 127: Automation options had been applied during AI Auto Play.

Flipped fraction in K-Mod's `CvCityAI::AI_getImprovementValue` had caused the AI to be *less* interested in Cottages when in Emancipation.

Fixed Civilopedia links to the Spy specialist article that had lead to the Spy unit instead. Similarly, links to coporations had lead to the HQ building instead of the coporation articles. Through helper functions `CvGameTextMgr::setSpecialistLink`, `setCorporationLink` and changes to some game text keys. *Credits*: Bug reported by crullerdonut (middle of the post)

Disregard non-rival units in `CvPlayerAI::AI_doEnemyUnitData`. (Because that data is used by AI governors for picking units that counter potential enemy units, e.g. Pikeman vs. Knight.) Seems like an oversight (but not a bug, strictly speaking).

When the game decides whether a starting site needs extra food (`CvGame::normalizeAddFoodBonuses`), Whale is no longer counted as an available food source (unless starting in Renaissance or later). BtS had failed to check the tech requirements of the necessary terrain improvement (Whaling Boats).

AI resource evaluation had not counted projects (e.g. Manhattan Project when evaluating Uranium) in production queues as currently relevant uses of the resource. Same bug with world units (which are not used by AdvCiv). These issue were introduced by the BtS expansion.

Fix counterintuitive rounding of hurry production costs that get reduced by a non-generic modifier, e.g. the settler modifier from the Imperialistic trait. *Credits*: Based on [this](#) CloseToHome Git commit

Call `updateDiplomacyAttitude` in the EXE after each update of the AI attitude cache that occurs while the Diplo screen is up. I think this aligns the leaderhead animations with the AI attitude level(?). Was previously only called after a civics or religion change (before updating the attitude cache), but there are several other interactions that can change relations values while the Diplo screen is up.

A K-Mod check in `CvPlayerAI::AI_espionageVal` had assigned 0 value to the Spread Culture mission unless the spy owner had at least 8% city culture in the target city. City culture is 0 unless a city has been previously owned, so probably tile culture was intended.

001b	(Not actually a bugfix.) Can't train air units in cities already filled with air units. (BtS allows them to be trained and rebases them upon completion, destroys them if that's not possible.)

When a newly completed unit exceeds a city's air unit capacity, the unit is moved to the city's rally point. If no rally point is set, the unit is moved (as in BtS) to the nearest city or fort with sufficient air unit capacity. The unit has its movement points spent at the start of the next turn. If there is no city or fort with available capacity, the unit is scrapped. When moving to the nearest city (no rally point) or scrapping, an on-screen message is shown. Clicking on the message centers the camera on the unit's new location.	It's impossible to move air units into a tile whose capacity will be exceeded, but cities without available air capacity are allowed to produce air units. Upon completion, an excess air unit is moved to the nearest legal city or fort or destroyed if there is none. The forced move doesn't cost any movement points. The city owner isn't notified about the move (nor of the destruction of a unit).

Config	Previously, I had made it impossible to produce air units in cities without available air unit capacity (in part, through change [064d](#)). That rule change can be a bit tedious when mass-producing air units in the late game, but it can still be enabled through `CAN_TRAIN_CHECKS_AIR_UNIT_CAP` in `GlobalDefines_advc.xml`.
Credits	Elkad made me aware that my old rule wasn't ideal with regard to usability. ([CFC post](#))
See also	[163](#) always spends the movement points of teleported units. That doesn't help in this case because units are produced at the end of a turn, just before movement points are restored.

001c	Display and calculation of GP birth probabilities

AdvCiv	*BtS*
GP birth probabilities are affected by the timing of GPP modifiers; e.g. specialists during a Golden Age contribute more to the probability of their GP type than specialists at other times do.	GP birth probabilities are proportional to progress values counted per GP type (no change in AdvCiv). GPP modifiers do not apply to those progress values.

Rationale	I wouldn't call it a bug, but it looks like an oversight to me. At any rate, it's counterintuitive when the per-unit progress values don't add up to the total progress value.
See also	Bug report, brief discussion on CFC: [link](#)

Corrected the birth probabilities shown on the GP bar (city screen and BUG GP progress bar). Now project the city's current per-turn GPP into the future.	Birth probabilities are based on the GPP collected so far; no projection. The probabilities adjust only gradually when a specialist is reassigned.

Tbd.	Should perhaps take into account a foreseeable increase of the GP threshold due to other cities finishing their GP earlier. And Golden Age length if currently in a Golden

	Age. That said, I might at some point change the way the threshold increases; better to leave the UI-side alone for now.
See also	078: Changes to BUG's GP bar options.
001d	In Debug mode, the Top 5 Cities and Wonders tab and Religion, Civics and Espionage screens were not properly switching to the perspective of another civ because of some errors in the Vanilla Civ 4 and BtS Python code. And the Top-5/ Wonders didn't reveal all information when the Info screen pops up at game end.
See also	007 deals with other changes to the Wonders tab in Debug mode.
Tbd.	Tech tree doesn't show the tech progress of the civ selected from the Debug menu; only shows which techs are known to that civ.
	The active player should be added to to the Debug drop-down menus first so that the active player is the initial selection. Currently, whoever is in player slot 0 is on top. (It's apparently not possible to make a selection programmatically.)
Credits	crullerdonut reported the issue with Top-5 cities at game end here (penultimate quote box).

Unrevealed cities are shown as "Unknown" on Top 5 cities (K-Mod: "Unknown" only if owner not met). Only revealed cities are considered for the list of high-culture cities on the Victory screen.

Rationale	Not really a bugfix on second thought but consistent with the K-Mod rule that unrevealed cities are secret.
001e	No more stop-trading requests about a civ that has just stopped being worst enemy; and no offers for Defensive Pact from a civ that has just been attacked.
Rationale	Because the AI can't immediately contact human players, the conditions for AI requests need to be checked again at the start of the human turn. This is handled outside the SDK and mostly works, but, apparently, some checks had been missing.
See also	134a deals with AI peace offers getting discarded by the EXE.
Tbd.	It would be good to check reasonable preconditions for all AI requests at the start of the human turn inside the DLL. Perhaps by moving the conditions that are checked before making each request (mostly in `CvPlayerAI::AI_doDiplo`) into subroutines. Rationale: (a) There could be further conditions that the EXE fails to check, and (b) the DLL could then reset the appropriate AI contact timer (since no contact was made); I don't think the EXE does this. The timer for embargo requests and DP offers are already reset by the DLL but only in the narrow circumstances described above (in the blue box). But what if the resolution of one diplo popup invalidates a subsequent one? At that point, the EXE has already fetched the entire contents of the diplo queue.
	(Update: There might currently also be problems in simultaneous-turns multiplayer with AI civs offering the same deal to multiple human players at the same time.)
	Preferable approach, probably less work too: Move AI-to-human diplomacy entirely to the start of the human turn and thus remove the need to double check the preconditions (and this would improve the quality of the offers too). I've written and archived (not published on GitHub) a draft for implementing this. And I've briefly posted about it here (second paragraph). Along with this change, an open issue with the timing of on-screen messages (see *Tbd*. under 106b) should be addressed. Problem: Peace proposals are currently made during the team AI turn as part of `AI_doWar`. I guess `CvPlayerAI::AI_negotiatePeace` and `AI_offerCapitulation` should be delayed until the diplomacy turn of the team leader. The parameters that CvTeamAI (or rather UWAIAgent) passes to those functions during the team turn are good; shouldn't have to be recomputed later on.

	Related "We the People" Git issue: link	
001f	Foreign cities no longer become unrevealed upon conquest by a third party	
When a city is conquered, it remains revealed to all civs that knew the city prior to conquest.		When a city is conquered, it is treated as a new city, revealed only to the new and former owner, and any third parties that happen to have visibility of the city at the time that it is conquered.
Rationale	Hard to say if this is really unintentional. The BtS code explicitly sets the city to be revealed to the former owner. Were third parties not considered? Doesn't make sense to me that third parties learn about the conquest but not about the aftermath.	
See also	I think, in BtS, the culture layer gives away the new city owner and the status of the surrounding tiles. [004z](004z) changes that.	
001g	Deleted duplicate `MemoryAttitude` entries about Suleiman and all leaders after him in `Civ4LeaderHeadInfos.xml` (it's ordered alphabetically); used the mean when two values contradicted each other. Those before Suleiman didn't have duplicates.	

001h	Industrial bad health	
AdvCiv		*BtS*
When a building leads to bad health with certain strategic resources (Factory, Coal Plant, Industrial Park with Coal or Oil), the AI assumes that the city already has these resources when evaluating the effect of bad health.		Only bad health that the city will suffer directly, i.e. from currently available resources, is taken into account. E.g. Factory counts as just 1 bad health so long as Coal/ Oil aren't available.
Starting in the Industrial era, so long as a city doesn't have power yet, the AI treats the city's current health as 1 less when evaluating the effect of bad health.		The AI does not aim at keeping a health surplus available; just aims at balanced health given the immediate effect of the building.
Increased the (negative) weights of bad health and food deficit in building evaluation.		
Hopefully no more (or only minor) population loss from bad health in Industrial AI cities.		AI ends up building Factories and Coal Plants before getting Oil and Coal. If health is just balanced at that point, once Coal and Oil are connected, the city is at -8 health. The AI then shrinks the population gradually by 8 to avoid wasted food (more with Industrial Park, not to mention Poisoned Water).
		BBAI and K-Mod have largely rewritten the building evaluation code, but this bit works as in BtS.
Rationale	Treating health as one less should lead to some leeway for later bad health, especially from power.	
See also	[160](160) makes Poisoned Water less dangerous, and [120e](120e) improves the AI response to that mission.	
Tbd.	The bad health effects are badly balanced. Too much at once, bad health for no reward (Factory doesn't actually get better with Coal or Oil), punishing for tall	

	strategies. Shouldn't even be possible to build a Coal Plant without Coal.

001i	Fog of war on routes	
AdvCiv		*BtS*
The pathfinder ignores unrevealed routes on revealed tiles, meaning that the waypoints shown on the UI can't give away fogged routes and that the AI does not count on using such routes. AI plot danger calculations disregard unrevealed routes on revealed tiles.		The fog of war hides tile ownership, units, cities, improvements and routes. The pathfinder treats all routes on revealed tiles as revealed. By plotting a path through fogged tiles, a player can learn whether routes have been built in those tiles.
See also	124 prevents the Trade layer from giving away routes in the fog of war. 182 prevents tile yields from giving away unrevealed resources on foreign tiles. 183 addresses some issues with cities and forts in the fog of war. 128 sometimes prevents the plot danger functions from cheating with visibility. advc.pf fixes other pathfinder issues. 004c fixes some minor issues with air missions leaking fogged info. 031 prevents recommended city sites from giving away foreign cities in the fog of war (by making the AI city site evaluation care about revealed tile owners rather than actual owners).	

001j	Deleted two (K-Mod) calls to `CvPlayerAI::AI_getNumTrainAIUnits` because `CvPlayerAI::AI_totalAreaUnitAIs` already counts those units.
See also	017 fixes a bug that also has to do with confusing these two function.
Added a couple of `isFriendlyTerritory` calls – had apparently been overlooked when BtS introduced vassal agreements (which allow passage even without an OB agreement). AI evaluation of Missionaries should be improved now.	
Replaced `ALWAYS_PEACE` option checks in AI code with calls to `CvTeamAI::AI_isWarPossible`.	
Rationale	When both `ALWAYS_PEACE` and `ALWAYS_WAR` are enabled, war wins out. The (Vanilla Civ 4) `AI_isWarPossible` function sorts that out and also handles `NO_CHANGING_WAR_PEACE`.
See also	105 replaces `getAnyWarPlanCount` with `AI_isFocusWar`; that's a vaguely similar change.

001k	AI plot danger no longer (indirectly) checks `isMadeAttack`	
AdvCiv		*BtS*
When the AI checks whether a plot is in danger of being attacked by a unit, it doesn't check whether that unit has already attacked this round.		Vanilla Civ 4 did not check for an earlier attack; BtS added this clause.
Rationale	Plot danger is only checked during AI turns, and the active AI civ then wants know if another civ will be able to attack on that other civ's next turn; it doesn't matter if the unit could *immediately* attack. Perhaps the BtS developers hadn't realized that the	

	`MadeAttack` flag persists for an entire round of turns.
	Hard to say how significant this bug is; the `plotDanger` functions are called in dozens of contexts. I don't remember how I noticed it; some situation in which the AI clearly underestimated plot danger.
See also	[128](#) doesn't always let the plot danger functions cheat with visibility. (Using the `canBeEntered` function from change [030](#) since v0.90.) No longer the case since v0.95. "More Naval AI" fixed this a few years earlier (though my latest fix is better): <u>Git commit</u>

001l	End combat upon reaching damage limit	
AdvCiv		*BtS*
Units with a damage limit below 100 hitpoints withdraw if they land a hit that reaches the damage limit exactly. E.g. a Cannon that lands its 4th 20-damage hit against another Cannon withdraws immediately after this hit.		Withdrawal happens only when the damage from a hit exceeds the damage limit. Therefore an unpromoted Cannon attacking another unpromoted Cannon, both at full health, will not withdraw upon the 4th hit and, if it can manage a 5th hit, that hit will apply 0 damage and withdraw. The combat odds are exactly 50%. However, the code that displays the combat odds assumes that withdrawal will happen upon the 4th hit, and so the displayed survival odds are 63.7%. The Advanced Combat Odds mod displays the correct odds; comment in the code (now deleted in AdvCiv): "*A catapult that deals 25HP per round, and has a combatLimit of 75HP must deal four successful hits before it kills the warrior - not 3. [...]*"
Rationale	Displayed and actual odds mustn't disagree. The logic for displaying the odds makes more sense here. A 0-damage hit is weird and won't appear in the combat log. It's also weird when the damage limit does nothing to make a combat less deadly; however, this is still the case for Artillery (damage limit 85) and is ultimately "just how combat works."	
See also	[advc.test](#): Code that uses simulations to compare actual odds with displayed odds. That's how I've found this bug.	
The hit that causes an attacker to reach its damage limit is included in the combat log.		The combat log only shows full hits.
Rationale	No reason not to show the final hit (now that it can't cause 0 damage anymore).	

001m	Scoreboard gets updated when a leader name changes	
AdvCiv		*BtS*
When the player enters a different leader name (Alt + D), the name is immediately updated on the scoreboard after clicking "OK".		The scoreboard isn't updated until end of turn unless the player manually closes and reopens the scoreboard.

001n	Some potential OOS bugs fixed. (I've introduced my fair share of these bugs, but only bugfixes in non-AdvCiv code are tagged with "001n". I'm also using that tag for some misc. code that I added for OOS debugging.)

The usual causes of out-of-sync errors in networked multiplayer are these:

(a) Local code (invoked by user input) modifies the game state. All uses of the sync'd PRNG (`SorenRandNum`) modify the game state. Some AI functions have a `bAsync` parameter that causes them to use the asynchronous PRNG (`ASyncRand`) instead. Also tricky: Functions that cache their return value, in particular `AI_baseBonusVal`, `AI_localDefenceStrength`, `AI_techBuildingValue` and `AI_obsoleteBuildingPenalty`, all at `CvPlayerAI`; `CvPlot::getFoundValue`, `CvCityAI::AI_buildingValue`, `CvCity::AI_neededFloating Defenders`, and the AI plot danger and closeness functions. Most have a `bConstCache` parameter to prevent the cache write. I think the cache writes wouldn't actually be a problem if the caches were reliably invalidated, but the formulas depend on so many variables that this is impossible to do.
(b) Access to uninitialized or non-allocated memory (e.g. out of array bounds)
(c) Use of `getActivePlayer` in global code (not invoked by user input)

(a) and (b) are also problematic in singleplayer. Rare, but something to reckon with: Sorting by memory address can cause OOS errors (K-Mod [Git commit](#)) and other errors (C2C [Git issue](#)).

Comprehensive guide by Gerikes about avoiding and debugging OOS errors: [CFC thread](#)
Overview by Afforess: [link](#)
CFC post by karadoc on the subject of OOS errors: [link](#)
Some posts by EmperorFool: [link](#)
By Nightinggale: [link](#)

See also	See [007](#) (logging), [135c](#) (Debug mode) and [127](#) (AI Auto Play) about multiplayer debugging. [kekm.27](#) adds a OOSLogger component See [003g](#) about potential issues with floating point arithmetic. Open K-Mod issue concerning (a): [link](#) (related forum [post](#)). I think [036](#) addresses this. [repro](#): Test for identifying non-deterministic code, addressing mainly error type (b).
Tbd.	`CvPlot::m_abBorderDangerCache` isn't stored in savegames and that seems a bit risky, though it might be fine. The AI plot danger functions cache some data. They're only called in synchronized code currently (except in Debug mode, and I've disabled these dangerous calls in networked games) as far as I can tell, but it would be nicer not having to worry about that, so a const-cache parameter or accessor should be added.
Config	I've left some code for debugging OOS problems commented out in `CvSelectionGroupAI::AI_update`. If an error is caused by the move of an AI unit, this will make it easier to identify the responsible unit. The `checkInSync` function could also be called from other (synchronized) parts of the code. The `bFullOOSCheck` flag in `CvGame::calculateSyncChecksum` should be set in addition.

001o	Main map blackouts

Changed the timing of a `CvDLLInterfaceIFaceBase::lookAt` call in `CvCity::conscript` that K-Mod had added. This seems to have fixed a bug that caused the main map to turn black and the unit pane (and field-of-view slider) to disappear upon conscription. That said, this problem did not exist in K-Mod, and it didn't occur in AdvCiv releases until v0.98. In release builds with global optimizations disabled, I've been able to trace the problem back to [this](#) Git commit, which is part of

v0.97. Breaking that large commit up, the decisive change is probably in the CvUnit class (or possibly CvPlot, CvCity). The main change to those classes is that the current tile of a unit or city gets cached as a CvPlot pointer in addition to being cached as a pair of coordinates. It's conceivable that there is indeed some problem with my implementation of that, but it's also conceivable that this change merely exposed some older problem, perhaps in an earlier AdvCiv commit, in the EXE – or maybe the K-Mod `lookAt` call was indeed erroneous, hard to say without knowing the exact semantics of that function. I write all this down (in Dec 2020) because I'm not sure if the problem is really fixed now. I've been experiencing rare, non-reproducible main map blackouts during AI Auto Play since at least Sep 2020, usually while I had the window minimized (but, then, I often minimize during AI Auto Play). In September, I had tried out a faster version of `Python24.dll` (see under [advc.make](#)). Reverting to the old version of that DLL later did not fix the problem, but it appeared to be more common with the faster DLL. Upd. (June 2021): I haven't experienced these random blackouts anymore.

If such problems persist, my best bet is to investigate `lookAt` calls. I've already tried calling `lookAt` with negative `x`, `y` and `z` coordinates. That (by itself) does not cause the main map to black out.

Credits	CFC user crullerdonut made me aware of the conscription issue: [bug report](#)

001p	Crash when loading a smaller game from inside a larger game
AdvCiv	*BtS*
Kill all cities early in the loading process. Check for valid leader type in `CvPlayer::getNameKey`.	The logic for loading savegames is partly outside the SDK, but the issue appears to be this: Cities remain untouched during the first stages of the loading process. As general game data and civs are being reset, the UI keeps getting updated, based on data about cities that are no longer consistent with the other data. This can result in a crash when the camera is near a city of a civ with an id greater than the highest civ id in the game that is being loaded (`CvCity::getMusicScriptId`) but apparently also in other circumstances (`CvCity::isVisible`). (Not sure what's going on with `getNameKey`.)
See also	[This](#) CFC thread seems to describe the bug. (No one posted a fix.)

001q	Safer definitions of integer type limits
AdvCiv	*BtS*
Define negative limits as signed expressions to make sure that they don't get treated as unsigned values. All uses of `INT_MIN`, `INT_MAX` replaced with `MIN_INT`, `MAX_INT`.	All limits are defined as hex literals. Only `MAX_INT` and `MIN_INT` are used. Vanilla uses those definitions consistently instead of `INT_MAX` and `INT_MIN` (`limits.h`). In BtS, a couple of `INT_MAX`, `INT_MIN` have snuck in, and K-Mod seems to have used predominantly (exclusively?) `INT_MAX`, `INT_MIN`.
Config	The definitions are in `CvGameCoreDLL.h`.
Rationale	The BtS constants are more than hazardous. For example,

	`int x = 0; return (x >= MIN_SHORT);` had returned `false`. Fortunately, no one before me used `MIN_CHAR` and `MIN_SHORT` (nor compared `MIN_INT` to an `__int64`).
Tbd.	Should probably get rid of those constants and use `limits.h` instead. It's confusing to have two sets of definitions and I don't see any benefit. Note that `std::numeric_limits` is not a good alternative because those functions can't be used in template arguments (not in C++03 anyway).

001r	Corrected AI computation of anarchy length in `CvPlayerAI::AI_doCivics`. (K-Mod bug)
See also	131 makes further changes to the switch-civics AI. I've posted about this bug on the K-Mod subforum: link (under No. 1, "Update")

001s	Some code that uses `canDefend` doesn't seem to take into account that all non-land units are defenseless against land units (yet `canDefend` returns true for all ships except Work Boat). One such oversight (in K-Mod code) causes the AI to ignore hostile units when stationing idle ships.
See also	139 further improves the AI code for keeping ships safe from land units.

001t	Preliminary fix for the following issue in K-Mod: After declaring war, an AI stack may decide to use a different path, even one for which no DoW would've been necessary. If this occurs, the AI now sticks to the original path (if it's legal) for at least one more turn so that a stack that triggers a DoW will immediately enter an enemy tile.
Rationale	The fix is more about concealing the problem than making the AI smarter; a DoW without crossing a border is an obvious mistake, sticking to a suboptimal path isn't.
Tbd.	I don't see how the DoW affects the best path; perhaps something in the `pathCost` function.Whatever it is, the pathfinder should anticipate that effect.

And a fix for a related bug: K-Mod sometimes lets stacks of non-city attackers move to an enemy city when war is imminent. Since these stacks aren't supposed to start the war, this doesn't normally make sense.

001u	Fixed: K-Mod code had failed to recognize units as obsolete when evaluating a strategic resource that it didn't have access to yet. This had sometimes led the AI to cancel and renew trades for strategic resources (especially Horse) every 10 turns.

001v	Bugs reported by vedg (Igor).
See also	046 and 905b address other issues (not bugs, strictly speaking) reported by vedg. He also contributed to one of devolution's bugfixes.

AdvCiv	*BtS* (presumably also broken in Vanilla Civ 4)
When a civ agrees to change both its civics and its religion in a single trade, all changes are applied immediately in a single revolution. The	Same if the religion change is added to the trade before the civics changes. Otherwise, only the

anarchy adds up.	civics changes are applied.
Rationale	See my reply to [this](#) post.
K-Mod bug: When a city finished a unit whose national limit was reached, overflow gold equal to that unit's production cost was paid.	
See also	[Thread](#) with the bug report. Obsoleted by [064b](#), which removes that part of the K-Mod code entirely.

001w	UI updates ("dirty" bits)
AdvCiv	*BtS*
When the happy or healthy status of a city changes through a resource trade, the indicators on the city billboards are updated.	Usually works – I guess the trade screen (not part of the SDK) does this? But not always. E.g., at least one time, an unhappy indicator appeared due to trades canceled by the AI and was not removed in response to subsequent trades during my own turn.
After a declaration of war, cached waypoint markers are updated.	After declaring war on a civ that the player didn't have Open Borders with, the cursor is shown in red when plotting a move into an enemy non-border tile. (At peacetime, this is correct – a move into a border tile triggers the declare-war popup.)
After entering or leaving WorldBuilder or Debug mode, the center unit of each tile (the one shown in 3D on the map) is updated.	When leaving Debug mode (also WorldBuilder?), flags of rival units sometimes remain visible on unrevealed tiles. K-Mod had already updated the flags ([Git commit](#)), but that's not enough.
Corrected the yield display on unowned tiles with a resource and 0 non-resource yield.	
Credits	CFC user ederl reported this problem [here](#) in the Realism Invictus thread (under "really small things" toward the end): "*Yields on neutral territorium seem not to be updated when ressources are revealed through technologies until I reload the game...*" The update works as far as I can tell, I think only tiles with 0 non-resource yield were affected.
AdvCiv	*BtS*
A bit of a hack: When a command button is clicked, the unit selection list is updated with a slight delay. As a side-effect, this gives the main map the mouse focus I think; anyway, it fixes the problem. K-Mod's Rapid Unit Cycling option also seems to fix it; therefore my bugfix is only used if that option is disabled.	When a unit command button is clicked, the button help text is often triggered during the unit cycling delay. It can then happen that the mouse focus remains stuck on the HUD and that the button help text remains on-screen. Hovering over a different HUD element resolves the problem.
See also	Based on [003r](#)
Tbd.	A similar glitch: When no unit is selected and a screen (e.g. Info screen, Trade screen) is closed by clicking (-I don't think the Esc key has this problem-) "Exit"/ "Farewell" immediately after opening it (perhaps before it's fully loaded/ displayed), the mouse focus gets stuck on the HUD. I expect that calling

	`CvDLLInterfaceIFaceBase::makeSelectionListDirty` from the exit handler of each Advisor screen (and somehow at the end of diplomacy) would fix this problem. Would have to expose that function to Python first.

001x	No extra Golden Age turn when finishing Taj Mahal during a Golden Age	
AdvCiv		*BtS*
When a building that grants a Golden Age is finished during the end of a player turn, the Golden Age turn counter is increased by the Golden Age duration at the beginning of the city owner's next turn. When the building is finished before the end-of-player-turn sequence, the Golden Age turn counter is increased immediately. For the Taj Mahal, this can only happen through a cheat command ([+] key), but, in a mod-mod, a special building like Academy might start a Golden Age.		Buildings that grant a Golden Age immediately increase the Golden Age turn counter with one turn added to the duration. As a consequence, if the building gets finished at the end of a turn, the added turn gets subtracted later in the end-of-player-turn sequence, but, first, the end-of-turn sequences of all cities are completed, i.e. some of the player's cities benefit from the extra Golden Age turn. Moreover, when the Taj Mahal is finished during a Golden Age, then a turn gets subtracted only from the ongoing Golden Age at the end of the player turn, i.e. a full turn is added to the total Golden Age duration.
Rationale	The full extra turn from Taj Mahal during a Golden Age is a widely known bug; see e.g. this CFC post. The other issues are also bugs in my book.	

001y	Fixed an issue with AI Work Boats being unable to find an order. BtS uses a "temporary hack" (in `CvSelectionGroupAI::AI_update`) to let such units skip their turn after 100 attempts to find an order. That hack is still in place as a fallback, and I intend to keep it permanently. I've reduced the number of attempts.
Added a another counter for the detection of and recovery from infinite loops in `CvGame::updateMoves`.	
Rationale	The code in `AI_update` can't handle infinite loops that involve units joining a different selection group.

001z	Fix Ctrl+H (select wounded units)	
AdvCiv		*Warlords* (the shortcut was added in Warlords)
When selecting units through Ctrl+H, wounded units with the same domain type as the currently selected unit(s) become selected. If there are none, Ctrl+H has no effect.		When units of different domains are in the same tile (city or fort), Ctrl+H selects only units of one domain that is chosen based on the internal order of the plot list, i.e. arbitrarily. If there are no such wounded units, then all units are unselected and automatic unit cycling (if enabled) proceeds to the next group.
Rationale	It's not possible to select units of different domains, so one domain has to be chosen.	
Credits	CFC user xyx made me aware of the bug here.	

002	Aesthetic changes
002a	Minimap shows lighter player colors than in BtS on water tiles (like in Military Advisor), and slightly more opaque colors on land tiles. No units are shown on the minimap.
Rationale	CFC forum post (also with screenshots). When Desert and Forest shine through too much, player colors can become hard to distinguish. Units: Can't make them out properly anyway, and can't tell how many there are because each stack is shown as a single blip.
Config	Options on the BUG menu (Map tab). Can also remove player colors from water tiles like in Civ 3. Until AdvCiv 0.99, these options were in XML.
See also	kekm.21 shows Barbarian territory on the minimap.
002b	Increased font sizes. Shortened some leader names, e.g. "Augustus Caesar"→"Augustus" when used outside of Civilopedia. Shortened "Native American Empire" to "Amerindian Empire". Added/ removed some tabs in Sevopedia's shortcut categories (004y) in order to repair the text alignment. Increased the total width of the civics screen (if the screen resolution allows it) and of the panels that describe the effects of the individual civics.
Rationale	People play on rather high resolutions nowadays, and the fonts don't scale properly. Long leader names take up (even) more room with the larger fonts. This is a problem for Native America because the text covers up the civics icons during diplo. Would be better to use only short descriptions there, but that seems to be programmed outside the SDK.
Config	The font-size change can be reverted by removing the `Resource` folder and `Assets\XML\Art\CIV4ArtDefines_Misc.xml`. This will also allow the mod to use any theme installed in `CustomAssets` (e.g. Blue Marble Gold). When a theme other than the AdvCiv theme is used, the `FONT_SIZE_FACTOR` in `GlobalDefines_advc.xml` should be adjusted so that the DLL knows how much text fits on the screen. The civics screen changes are all in `CvCivicsScreen.py`. It's easy to make the panels in the upper half wider as well; I went out of my way to avoid that because I don't think it looks good.
See also	061 (improved help text for listing the units in a tile) is the only DLL code so far that adjusts the amount of displayed text to the font size.
Credits	Inspired by VIP mod and I also took a look at vincentz's setup. This tutorial was helpful, although neither the downloadable package nor the code in the tutorial actually worked for me. This post by CFC user hagnat helped me fix some Vanilla Civ 4 and BtS errors in the theme files, though I don't think these errors were actually causing any problems. They did lead to confusing log output. See also this conversation on CFC. (In the thread, I conjecture that the log only gets created upon encountering a critical error. Not true; I just wasn't aware that it gets created in the BtS install directory.)
Tbd.	Can't seem to change font sizes without defining a custom theme, which appears to mandate copying 25 thm text files. I was at least able to avoid copying all the tga files, which make up 15-30 MB – this

would've been prohibitive for distribution.

I think there may be a way to make this work by copying just one or two files, but I can't figure it out (and I don't think other modders have).

Had to hardcode the mod folder name in `Assets\XML\Art\CIV4ArtDefines_Misc.xml` in order to plug in the custom theme.

Could I verify before the mod tries loading the theme whether the game is installed in the correct location and display a more meaningful error message than "failed to initialize primary control theme"? perhaps helpful for this: C2C Git issue

To allow users to rename the mod folder, I could probably create a file with the contents of `Civ4.thm` (but using the current mod folder name, whatever it is) in the Windows temp folder and change the path of `DEFAULT_THEME_NAME` (Misc. ArtDefine) to that temp file before the theme gets loaded.

Allowing the mod to be loaded from `My Games\Beyond the Sword\Mods` seems tougher. Somehow, the .thm files would have to acquire the paths to the Vanilla and BtS folder dynamically (replacing the `\..\` path prefixes that blindly navigate two or three folders up).

AdvCiv	BtS
Tile help text shows aggregated information (e.g. "Archer (2)") about units if there are more than 12.	The threshold is 15.
When info is aggregated, each unit type gets a separate line.	Sometimes tries to put them all in one line, which leads to chaotic linebreaks.

Rationale	Larger font means there is space for fewer lines and less space per line.
See also	061 overhauls the unit list in tile help text and also takes into account the font size; the above only applies if those changes are disabled.
	092 (incomplete) is supposed to increase the size of the unit pane. That should also make the area for help text wider. There seems to be no direct way to make that area wider – `HELP_TEXT_MINIMUM_WIDTH` doesn't do the trick. The "We the People" modders seem to have run into the same problem: Git issue
002c	Added a couple of translations for K-Mod game text entries.
Credits	Also merged translations by Zholef. I haven't marked these changes in the XML files.
002d	Changed the icon of the specialist-commerce ability introduced by K-Mod 1.45 to show a Scientist; was Citizen. (Depending on which commerce type gets increased, the icon could also show a different specialist, e.g. Engineer for production.)
Changed the icon of the Ocean trade effect (Astronomy) to the one that had been used in Warlords.	
Credits	crullerdonut made me aware here (under the penultimate quote box) that the Coast trade and Ocean trade effects use the same icon, and that the proper icon is available among the art assets.
002e	Disabled glow effect on units ready for promotion except for the currently selected unit and foreign units.
Config	"Unit Icons" tab of the BUG menu. (The "Promotion Available" option was not added by me; that's part of BUG's "PLE" component. While I don't want to enable it by default, that option seems sensible to me.)
Rationale	Too obtrusive. Players don't select units just to assign promotions; they select them in order to move or heal, and assign promotions at that opportunity. No need to

	constantly indicate unassigned promotions on the world map.
	The glow is needed on foreign units because there is no other way of telling that a foreign unit has unassigned promotions. Shouldn't be a secret either. (Rarely relevant in singleplayer as the AI always assigns promotions right away.)
002f	Changes to city icons: Defense icon moved behind the defense modifier; city network icon removed. Airport icon merged from BULL and enabled by default.
Rationale	Should be easier to guess now that the tower icon isn't some status indicator but just stands for the word "defense".
	The city network icon is at best helpful in the early game, though probably not at all for experienced players (and inexperienced players may be unaware of the icon). Can always look at the list of trade routes on the city screen or consult the Trade layer.
	I haven't played with the airport icon yet, but it sounds quite useful.
Config	Option for the city icons on the "Map" tab of the BUG menu. The airport icon is in the `GameFont.tga` I copied from BULL. (BULL does not have a custom `GameFont75.tga`, but BUG and K-Mod do. Looks like that one's needed for the smiley option of the Glance tab.)
See also	076 disables the icons placed by the "Detailed City Info" option. 101 adds, optionally, an icon for cities with positive revolt probability. 187 adds the airport icon to the 75% game font file so that it can be used in hover text.

Option on the "Map" tab of the BUG menu for re-enabling the "Detailed City Info" icons along with a new icon for highest espionage rate and highest free XP. And another, similar BUG option: icon at the city with the next projected Great Person birth. All disabled by default. See the hover text for documentation. The icons are shown when the active (human) player has at least 2 cities; BtS had required 3.

Config	Originally, I had implemented a separate option for each icon. Those individual options are still functional on the DLL-side, but commented out in XML and Python. Also left commented out: Options for icons at the Globe Theater and National Park city.
Credits	Brief discussion with crullerdonut on CFC: 1 (middle of the post) 2 (start of the post)
Rationale	It was easy to do and someone will always miss any established UI feature. As for the new icons, it remains to be seen if anyone will find them useful. The only I could see myself using (when playing without the GP progress bar) is the GP icon at the city where the next GP will be born.
	I've though about tying the icons to national wonders. That might be a bit more useful, but would be pretty haphazard as there isn't an appropriate icon for every wonder, some of them don't need an icon at all, and Scotland Yard isn't a national wonder (but a good place for showing an espionage icon). Would also not work well in mod-mods with different national wonders.
	There are still some unused icons left that could possibly make sense: The combat strength icon (perhaps for the highest military production rate – if it's not the same city as the one with the highest generic production rate), the negative gold icon (for cities with very high maintenance), the culture icon (for cities that somehow look relevant for a culture victory?). All a bit too complicated …
	The combat strength icon could also replace the Great General icon for the city with the most free XP. The GG icon is difficult to see – but the combat strength icon also isn't easy to see and is less fitting.

Another city bar icon: Spoiled food icon shown when a city is set to "Avoid Growth" (AG). And	The tiny button on the city screen being highlighted is the only reminder about AG. The

some other UI reminders about that setting: When a city has a food surplus despite "Avoid Growth", print "Avoiding Growth" on the city screen's food bar and show the turns-to-grow (i.e. turns until food gets discarded) in red. On the main map, show the gray status indicator (behind the population count) in that case. Show a red indicator when the city will discard food at the end of the current turn. Mention AG in the city bar hover text when enabled. Added a loding-screen hint about the city bar icon, along with a reminder about turning on citizen automation during AG.	food bar on the city screen and on the main map looks completely normal, as if the city were growing (if there is a surplus). When citizen automation is enabled, AG will usually cause the governor to reduce the food surplus to 0; the governor may even shrink the food store.

Rationale	AG is so easy to forget that it's almost not usable. The uses of AG without citizen automation are pretty limited in any case: A player who doesn't want a city to grow will normally manage to reduce the food surplus to 0 through citizen reassignment, and then there is no need for AG. That said, it can happen that the assignment with optimal non-food output will still have a small food surplus. Important to make the city bar icon optional because players may use AG to permanently keep a city from growing (i.e. with citizen automation enabled and resulting in 0 food surplus), and, in that case, the icon can be more annoying than helpful.

AdvCiv	*BtS*
The last of the three icons contained in BULL's `GameFont.tga`, the Citizen icon, is used as the heading of the "population" column of the (non-customizable) Domestic Advisor screen.	The population column heading says "POP". All other columns are headed by icons. I guess BULL uses the Citizen icon only for city bar hover text.

See also	004 uses the strength icon for the stationed-units column.

002g	Changes to Main Menu (the one that opens right after launching BtS)

AdvCiv	*BtS*
In-game main menu is referred to as "Game Menu". Shown grayed out: "BtS Content", "Check for Updates" (in the "Advanced" submenu); "Internet Games" (in the Multiplayer submenu).	"Main Menu", same name as the menu where you go after "Exit to Main Menu." "BtS Content" leads to an empty screen. The Update site – apparently hardcoded as http://motd.gamespy.com/motd/vercheck.asp in the EXE – has long been disabled; hangs for some seconds, then reports that the game is up to date. The GameSpy platform for finding opponents on the Internet has been shut down in 2014.

Rationale	Can't remove the dead items (handled by the EXE), so it seems that the best I can do is show the text in gray. I had "Play Now! and "Play Scenario" grayed out at first, but "Play Now" shows descriptions of map scripts that aren't visible elsewhere and there's a preview screen for some map types and scenarios (though a pretty bad one). Could rename "Custom Game" to "New Game (Randomized Map)", "Play Now" to something like "World Picker" and "Custom Scenario" to "New Game (Scenario)", but I think players are so used to selecting "Custom Game" that this would be an

	inconvenience.
	Might be nice to suggest a different way to find human opponents. There's only CFC though, which players will already be aware of. Steam has its own lobby, but my understanding is that players using Steam's GameSpy version can't go there and that AdvCiv (as well as most – if not all – mods) requires the GameSpy version.
Tbd.	Perhaps rename "Exit to Main Menu" to "Exit to Opening Menu" – that's what it's called under graphics options. Could then replace "game menu" with "main menu" again.
	Perhaps refer to the main interface as the "Main Screen" in game text as it's done in the Civ 4 manual; "interface" is slang.
See also	Kek-Mod has an in-game update function (inherited from PBMod I think): CvModUpdaterScreen.py

002h	Changes to textures
Config	Remove `Assets\BML.fpk` (renaming the file doesn't help) in order to use the BtS textures, or put a different `fpk` file in `Assets` (e.g. the Blue Marble texture pack linked in the opening post of the K-Mod thread (last paragraph). If BtS textures are used, clouds should probably be re-enabled through `RENDER_GLOBEVIEW_ CLOUDS` in `GlobalDefines_advc.xml` because that switch can't remove the shadows.
	To modify my textures, `BML.fpk` has to be unpacked first, using PakBuild.
Credits	I've worked from textures in the Blue Marble (BM) design.

AdvCiv	BtS	BM
Changed the brightness, saturation and color balance of all terrain textures except Hill, Snow and Peak, aiming at a middle ground between BtS and BM. I think my colors are less saturated than either BtS and BM. Coast is quite a bit brighter than Ocean.	The BtS textures have bright colors. They're easy to distinguish except perhaps Coast and Ocean.	Much darker colors than in BtS; Coast and Ocean somewhat hard to distinguish.
I've also adopted the water-related non-terrain textures from BM to my color scheme.		Also changes several non-terrain textures that depict water (e.g. irrigated farm) to match the new color of water.
Dark green Forests, medium green Jungle, light green Grassland.	Bilious green Grassland and Jungle, medium to dark green Forest.	Dark green Forest and Jungle, medium green Grassland that contrasts very strongly with river banks.
No changes to the texture detail, though reliefs might be more visible due to the lower color saturation.		More detailed texture reliefs. Makes Grassland in the fog of war and non-fogged Jungle a bit hard to distinguish.
Clouds disabled.	When zooming out far, passing clouds are shown; these clouds also cast shadows.	No shadows; denser, more realistic clouds.
No change to textures outside the map.	Some textures resembling those on the map are shown in	Adjusts these textures to match the BM style. (Compressed size:

Adopted the BM resource textures, and also toned down Corn, Wheat and Silk. Roads light brown. Railroads from BM but with increased contrast for a dotted look.	Civilopedia and the "Play Now!" World Picker screens for illustration. Roads dark beige, Railroads light gray, difficult to tell apart.	6 MB) Less garish textures for Banana, Rice, Gems and Uranium. Darker Railroads, no change to Roads. If BUG is used in addition, the I Love Asphalt component (not included in K-Mod/AdvCiv) colors roads black.
Rationale	I mostly wanted something that's easier on the eyes. Blue Marble is pretty good in that regard, but has some other issues (see above) and looks more different from BtS than it has to. I've also gotten the (compressed) file size down to 5 MB from 15 MB. The BM reliefs look kind of nice but more crayon-like grainy than detailed. All three versions let Coast bleed far into adjacent Ocean, which can make it hard to tell where the Coast ends. Probably can't be helped because the player is – apparently – supposed to be able to tell when there is an unrevealed Coast next to a revealed Ocean.
See also	CFC post with screenshots comparing the AdvCiv, BM and BtS textures. Water textures by CFC user MightyToad (who I think is very capable at what he does). Those look good when zooming in, but, at the normal playing distance, the waves are too big; not what the seas look like from the stratosphere. I've made a version with a 2x2 mosaic of one of MightyToad's textures (and also with some color adjustments) and that does result in smaller waves, but I guess they're still too big; anyway, it still looks too detailed overall, perhaps especially since the land textures aren't similarly detailed.
Tbd.	In the fog of war, the deciduous forests in the subtropics are too difficult to distinguish from tropical jungle. Making the forests darker or less reddish doesn't help. Brighter jungle would help, but would look too unnatural. (That's what BtS did.) Sparser forests? Don't really like that either. May have to apply to all forest varieties and also to routed forest, which already looks too sparse. Related CFC post (near the middle) Update: I've made jungle a bit brighter again; not sure how much it helps. Natural jungle can look pretty bright; see e.g. this photo on Wikipedia.

002i	Player color palette revised

See table below. The BtS palette is on top and the AdvCiv palette on the bottom. The empty cells happened by accident. The colors labeled as unused are only used as secondary colors (when multiple leaders of the same civ are in one game), and light black (lower right corner) is entirely unused. I've been less careful about making these colors distinct than about the primary colors.

Made a few colors like the Aztec's green (upper left) and the Persian's blue (top center) less saturated and thus hopefully less painful to look at.

Most colors shifted a little bit to make them easier to distinguish.

Barbarians	America	Mongolia	Persia	France	Korea	Ottoman
		Portugal	China	Rome	Russia	Inca
Germany		Carthage	Mali	Spain	India	Japan
England		Arabia	Greece	Zulu		Celtia
Babylon	(unused)	Byzantium	Ethiopia	Holy Roman	Khmer	Maya
(unused)	Native America	Netherlands	(unused)	Sumeria	(unused)	(unused)

Barbarians	America	Mongolia	Persia	France	Korea	Ottoman
		Portugal	China	Rome	Russia	
Germany	Aztec	Carthage	Mali	Spain	India	Japan
England	Egypt		Greece	Zulu	Viking	Celtia
Babylon	(unused)	Byzantium	Ethiopia	Holy Roman	Khmer	Maya
(unused)	Native America	Netherlands	(unused)	Sumeria	(unused)	(unused)

Config	Remove `Assets\XML\Interface\CIV4ColorVals.xml` to restore the original colors.
Credits	Andy Langton's [Hex RGB Colour Converter](#) came in handy.
See also	[002a](#): Increased opacity on the minimap makes the colors a bit easier to distinguish there. duckstab's [btsColorEditor](#) with its "Adjust" button could've been useful if I had known about it.

Also made the green of the research bar, the orange of the food bar a bit darker and the yellow of the GP bar darker and greener.

Rationale	The research bar was unpleasantly bright and too similar to `COLOR_POSITIVE_TEXT`. White text was difficult to read on the bright food and GP bars. City names on the main map are still not easy to read.

AdvCiv	*BtS*
When the same civ is used by multiple players (e.g. Washington, Lincoln and Roosevelt in the same game), then color schemes (primary color, text color, secondary color) from unused civs are used for all players beyond the first: For the second player, a civ with a primary color resembling the original civ's secondary color is chosen; e.g. Lincoln as the second American leader in a game will receive England's colors because the secondary color of America and the primary color of England are both white – that is, unless England is also in the game. For the third	Color schemes from the back of `Civ4PlayerColorInfos.xml` are chosen. The last few of those color schemes are normally unused, but only in the sense that the color combinations are unique; the individual colors aren't unique, and can easily clash with the default colors of other civs in the game.

player, a civ with a primary color resembling a mix of the original civ's primary and secondary color is chosen; e.g. Roosevelt as the third American leader will receive a primary color between middle blue and white (the Viking colors in a test, but perhaps that was because the Greek colors were already taken). Leaders beyond the third (only possible in a mod-mod) receive the color scheme of an unused civ chosen at random.	
See also	[Screenshot](#) posted on CFC that show clashing colors.
Rationale	There are a few color definitions that are unused or used only as a secondary color. To improve on the BtS approach, those available colors, perhaps along with some novel color definitions, could've been used for a reserve of colors that can't clash with other primary colors. However, I think it's nicer to have unique replacement colors, i.e. chosen based on the unique (well, mostly) secondary color. For some civs, that secondary color is even quite fitting as the primary color, e.g. red for England. Using the color scheme of another civ was easier to implement than to create a color scheme around the secondary color dynamically. Choosing a color in between the primary and secondary color (for the third player of a civ) results in a color that is fairly easy to distinguish from the other two and establishes a color range that ties the three players together.

002j	Changed the sizes of a few 3D models, especially ships.
Rationale	The sizes had already been scaled by hand, but a few units appeared oversized, especially Modern Work Boat. (Of course, all units are kind of oversized, and I've tried making them all 20% smaller, but that made them significantly harder to distinguish.)
Config	Through `Art\Civ4ArtDefines_Unit.xml`
Credits	Inspired by Elhoim's [Better Ship Scale](#) mod, though that mod makes the sailing ships look like nutshells, and the modern ships poke across the tile boundaries.
See also	[905b](#) increases the size of East Indiaman a bit because it's supposed to be bigger than Frigate.
Tbd.	Building scales could probably also be improved. E.g. the Taj Mahal is a bit small ("Dawn of Civilization" [enlarges](#) it). But I don't want to copy `CIV4ArtDefines_Building.xml` from BtS until I have more ideas what to change. Unit group sizes: Machine Gun and Gunship might look better as groups of two. For Bear (idea from the [PAE mod](#)) and Panther, group size 1 would make much more sense. Not sure about all the mechanized land units having group size 1; size 2 might be better. However, such changes aren't just a matter of UnitMeshGroups in `Civ4UnitInfos.xml`; the units also would have to be rescaled in `Civ4ArtDefines_Unit.xml`, and other adjustments might be needed in addition. Gunship rotors in frozen animation blot each other out. It seems that a larger group of Guided Missiles would require a change to the Guided MIssile attack animation.

002k	Put "AD" (Anno Domini) before the year number in all game text. And removed the colon after "Turn" on the time display.
Rationale	I understand that "1500 AD" is acceptable, but "AD 1500" is more proper.

Config	Option on the "Time" tab of the BUG menu. Can also pick CE/BCE there. Or edit the text keys in `CIV4GameText_advc.xml`.

002l	Tweaks to sound effects	
AdvCiv		*BtS*
Play combat audio based on player's era.		Based on the game era.
Rationale	I haven't checked, but, <u>apparently</u>, the sound differs for each era, and I think all era-based visuals depend on the (human) player's era, so it seems inconsistent to base the audio on the game era.	
No sound is played when a unit takes nonlethal Flank damage, and the respective message is displayed immediately.		The victory or defeat sound is played, and the message appears with a slight delay for the attacker and, for the defender, only at the start of the next turn (`bForce=false`).
		All(?) the other combat-related message use `bForce=true`.
Rationale	Showing combat messages directly is generally preferable, but multiple messages with an associated sound can get noisy (constructive interference I suppose). Nonlethal damage isn't significant for a sound anyway (at least not for the victory and defeat sound); for units killed by Flank damage, I'm keeping the sounds and `bForce=false`.	
See also	<u>CFC post</u> by me on this subject	
When multiple human units are promoted with one click, the promotion sound is played only once.		Played simultaneously for each unit, i.e. can get loud.
When a human stack makes a Stack Attack, a victory or defeat sound is played only for the final attack.		A sound is played for each individual attack, all on top of each other.
Rationale	If the final attack fails, then the defeat sound is appropriate even if the majority of the earlier attacks succeeded – because the attacker lost at least one unit and hasn't managed to eliminate all defenders. If the final attack succeeds, then the victory sound is not necessarily appropriate, but there isn't a familiar neutral sound, and at least the current behavior is simple and somewhat intuitive.	
Tbd.	There might be a couple more cases of stacking sounds to take care of. Looks like the "your units are under attack" sound could be a problem.	
	Would be nice to have a general safeguard against loud sounds. I think all sounds get played through either `CvDLLInterfaceIFaceBase::playGeneralSound` (one version of that function is unused; can simply be set to `protected` visibility) or `addMessage` with `bForce=true`. Could insert a call to a safeguard function at CvPlayer (or at a new component of CvPlayer) into those two functions. The safeguard function would keep track of which sound has started playing when and how long it is expected to last, and would return `false` when too many sounds overall or multiple instances of the same sound are about to get played.	
See also	<u>CFC post</u> by me about promotions and Stack Attack.	

The war-horns sound is only played for primary declarations of war.	The multiple (pairwise) DoW at the start of a war involving vassals or defensive pacts result in a single loud war-horns sound. Sounds at the end of a war can also stack when vassals are involved.
Play the deal cancellation sound only once when declaring war.	Declaring war cancels all deals. A sound is played for each canceled deal at the same time.

See also	106o combines announcements of declarations of war and peace deals that involve vassals into a single message – for which a single sound gets played. (Before AdvCiv 1.04, change 002l had suppressed the sounds for war/ peace changes of vassals; that approach might be easier to merge into other mods.)
	106j plays no deal cancellation sound when the AI decides to cancel (and renegotiate) a resource trade deal.

When multiple Global Warming events are reported at the start of a turn, a sound is played only for the first one.	One sound per event.

Credits	crullerdonut made me aware (2nd quote box): CFC post
Rationale	I don't think the sounds had stacked up, but it's still annoying to hear the same sound six times in a row.

When a sea unit withdraws from combat, one of the water movement sounds is played (at a slightly increased volume).	The marching-boots sound is played for all withdrawals. (Sea units can withdraw through promotions.)

Credits	From the Dawn of Civilization mod: Git commit, related post (But I use a slightly different – less harsh – sound.)

002m	Shortened unit animations
AdvCiv	*K-Mod*
The base number of animated combat rounds decreases slightly with each era. As a result, the animated units die after fewer hits in the late game than in the early game, and combat animations per battle take about 30-50% less time in the late game. Moreover, on the defense, starting in the Renaissance era, combat animations are cut short, meaning that they usually stop before one side has been defeated. In addition to the above, when playing with the "Single Unit Graphics" option (SUG) in single-player mode, the base number of combat rounds is halved.	From the K-Mod changelog (v1.29b): "*Combat animations are now orchestrated to roughly correspond to the events from the actual combat mechanics. (It still isn't a blow-for-blow representation of the actual battle, but events in the animation will at least occur in the same order as the events in the combat log.)*" That is, the number of animated combat rounds is based on the number of entries in the combat log. In addition, it's multiplied by an XML-configurable base value. Looks like SUG doesn't affect the number of rounds, meaning that the two men are shown hitting each other for about 10 seconds.
Config	`GlobalDefinesAlt.xml` (because that's where K-Mod's `STANDARD_BATTLE_ANIMATION_ROUNDS` is defined)

Rationale	For attacks by the player, there's the "Quick Combat (Offense)" option, but "Quick Combat (Defense)" makes attacks against the player difficult to follow, and animating them takes too long in the late game.
	Sadly, the speed at which the animations are played can, apparently, not be modified. I think the EXE consults functions like `getSecsPerTurn` in `CvDLLUtilityIFaceBase.h` for this. (`CvUnit::getAnimationMaxSpeed` is called during combat but doesn't make any appreciable difference.) Can only speed animations up by making units die after fewer hits or by ending animations before one side has been defeated. The latter doesn't look nice, but the former just isn't enough when large AI stacks attack in the late game. And players still have the option to see fully animated combat on the attack.
	Fewer hits per kill as the game progresses could be interpreted as weapons becoming deadlier.
	SUG: It looks goofy when a single Rifleman takes bullet after bullet without going down. I'm applying my change only to singleplayer mode because I'm not sure if basing the combat rounds on player options could lead to synchronization problems in (Pitboss) multiplayer games.
See also	Similar arguments in this CFC thread. Two users claim that SUG speeds up combat – perhaps it does in BtS; not in K-Mod though.
Tbd.	Could I just show the eventual victor knocking down all opposing figures that remain with one strike a piece when combat is cut short? Or approximate the combat rounds more coarsely, e.g. if the sequence of per-round wins in a fight A vs. B is ABBABBAAABA, show animations for ABABA instead. The algorithm could be to choose a target number of successes for the victor (at least as many as there are figures on the losing side) and, based on that, a smaller target number of successes for the loser. Then somehow pick a sequence that resembles the original sequence; perhaps start by preserving the true first and last hit ...

AdvCiv	*BtS*
Shorten the air mission animation for recon, strike and bombing to 4 seconds, i.e. play them faster.	5 seconds each
Rationale	The long animation times aren't really a problem as the missions take effect immediately and don't block the UI. (And the air strike animation isn't shown at all when playing with Quick Attack.) Still a bit distracting to see the aircraft hover for so long. Faster aircraft look also more realistic. Perhaps a bit easy to miss for a first-time player, so it's understandable that Firaxis made the animations so slow.
Config	`Civ4MissionInfos.xml`. The `iTime` value gets taken times 250 ms (`CvDLLUtilityIFaceBase::getSecsPerTurn`).
Shorten the nuke animation to 8 seconds.	10 seconds
Rationale	Still very long, but, if players actually want to see it, – it's not going to look good when played much faster.
See also	650 deals with other changes to nuclear war.

A 2-second nuke animation, i.e. mostly a camera rumble plus explosion sound (if zoomed in close enough), is shown when Particle Effects are disabled. 1 second when playing with simultaneous turns and for enemy nukes. When the full animation plays, the UI will still (mostly) be locked.	Always 10 seconds, but, without Particle Effects, no explosion is actually shown and, on enemy turns, the camera doesn't focus on nukes. With simultaneous turns, it looks like the animation will play for all human players regardless of whether they can see it; but I haven't tested it. In contrast, the "Show … Moves" options are treated as disabled when playing with simultaneous turns. While the nuke animation plays, automatic unit cycling is disabled and only units in the same tile as the nuke unit can be manually selected.

Credits	crullerdonut made me aware that the nuke animation requires Particle Effects here (end of the post).
See also	The end of my previous post (link above).
Rationale	The best solution might be to stop the animation from interfering with unit cycling. More specifically, it seems to be the "delayed death" status of the nuke unit that inteferes. It's easy enough to kill the unit immediately, but that also cuts the animation off. There's probably a way to allow cycling while the selected unit is dying (and the animation playing), but it's too much effort for me to figure this out. 2 seconds is still kind of long when there is no animation to see, but nukes have a recon effect during their animation, and player may want to use that opportunity to scout out enemy unit positions. As much as that illogical recon effect stinks, – wouldn't want players to enable the full animation just so that they don't lose that benefit. I don't think an animation can be shown without active visibility. Simultaneous turns: Tedious to test, so I want to keep it simple. Apparently, playing an animation that blocks the UI on only one machine would be quite a handicap (not sure about synchronization).
Config	BUG option for showing the short rumble or no animation at all regardless of the "No Particle Effects" option.
Tbd.	Would be nice if "Show Enemy Moves" could move the camera to the impact sites of enemy nukes. But, even then, I don't think the full animation should be shown; should suffice to show that when the active (human) player detonates a nuke. Perhaps OK to remove the recon effect, i.e. never to play the animation when dropping a nuke in the fog of war. Players could still marvel at the animation when they nuke a visible tile (not so uncommon – how else can they target enemy unit stacks).

Show the main screen message ("... launches a nuke ...") immediately (`bForce=true`) when using the shortened animation.	The message is shown immediately only for nukes fired by the active (human) player. Otherwise at the start of the next turn.

Rationale	Easier to follow enemy launches this way.
See also	004g shows bombardment messages immediately.

002n	Flashing end-turn message disabled	
AdvCiv		*BtS*

No flashing message is shown at the end of a turn. No change to the animation of the end-turn button, and no change to flashing reminder messages.	Once all units have orders and all popups have been dealt with, the end-turn button starts to pulsate and a flashing message "Press enter ..." is shown until the player ends the turn. The Reminders mod component (Alt+M) replaces the press-enter message with the reminder message set by the player.
Config	Can be re-enabled on the "General" tab (column "Misc.") of the BUG menu.
Rationale	Helpful for first-time players, but a little annoying in regular games, and quite annoying when taking screenshots. Newbies should still be able to notice the (subtly) animated end turn button eventually.
See also	004t disables the flashing exit-city-screen message. 106l suppresses the Autosave message.
Tbd.	In games with simultaneous turns, the "waiting for you" message seems to get shown as soon as all other players have ended their turns. That's a bit annoying, though I guess some indication is necessary (one time announcement?). Also the end-turn button turns red at that point – that should never happen when there are still units that need orders (`CvPlayer::hasReadyUnit`). This stuff is in part handled by `CvGameInterface.cpp`, e.g. `CvGame::shouldDisplayWaitingYou`.

002o	Don't play the same music track twice in a row
Rationale	It would generally be better to shuffle the tracks upon entering a new era and upon loading a savegame. But not playing the same twice is at least an improvement and was easy to implement. (Actually, it looks like the original developers had meant to implement this.)

002p	No Aqueduct graphics across Coast	
AdvCiv		*BtS*
Tiles in the radius of any city with an Aqueduct can be the starting point of an Aqueduct graphic only if they're on the same continent as the city.		The Aqueduct 3D model starts in some Peak, Lake or Hills tile within the city cross. The EXE selects that tile based on a function `CvPlot::getAqueductSourceWeight` in the DLL. The city where the Aqueduct is located is not indicated to the DLL function, and the EXE does not check whether the source tile and the city are on the same continent.
See also	Example of a goofy looking Aqueduct: CFC post	
Rationale	Maybe the DLL can figure out exactly which city `getAqueductSourceWeight` is being called for by keeping track of earlier EXE-to-DLL calls, but this simple solution seems good enough. Will sometimes (rarely) rule out Aqueduct sources unnecessarily when radii of cities on different continents overlap.	

003	Style changes and other refactoring, utility functions, comments about unused or otherwise dubious code; in particular:
Minor refactoring changes to improve readability in large parts of the C++ code base:	

Variable declarations moved to the point where the variable is initialized; `continue`, `break` and `return` statements to reduce indentation; variables named `bIsSomething` renamed to just `bSomething`; removed (obviously) unnecessary parentheses; made comments more compact, in particular those by jdog5000; removed blank lines and curly braces around one-liners unless the braces are needed to separate the one-liner from a multi-line condition; switched conditions like `0!=x` around to `x!=0`; array initialization loops replaced with single-line initialization; `INT_MAX` and `INT_MIN` replaced with Civ's `MAX_INT` and `MIN_INT`, breaks in long lines (ca. 85 to 100 characters). removed the `xmlKey` parameters from `getBUGOption`... calls because the callee doesn't use them; `const` qualifiers added. For what it's worth, the last few Firaxis programmers working on the codebase (patch 3.19, Colonization) had been making changes (mostly localized) in a similar vein.

Those changed aren't marked with comments. Structural changes are sometimes tagged with an "advc" comment, but I've started deleting those comments again. In particular, `continue` statements without a comment can be assumed to come from AdvCiv.

Macros `TEAMID` added, and `GET_TEAM` can now also take a `PlayerTypes` parameter. For example,
`GET_TEAM(GET_PLAYER(ePlayer).getTeam())`
becomes
`GET_TEAM(ePlayer)`
(akin to the functions in BUG's `PlayerUtil.py`).
Macro `PLAYER_TEXT_COLOR` added to complement `TEXT_COLOR`; mainly used in `CvGameTextMgr.cpp`.

Shortened `gDLL->getInterfaceIFace()` to `gDLL->UI()`.

Added some free `getActivePlayer` functions (local to implementation files) in order to abbreviate `GC.getGame().getActivePlayer()` in implementation files that need to access the active player (or team a lot). Also added boolean member functions `isActive` to CvPlayer and CvTeam and `isActiveOwned`, `isActiveTeam` to all classes that can be owned by a player or team.

Moved a lot of code out of `CvGameCoreUtils.h`/`cpp` – to more specific places, in part to new translation units.

Rationale	It's often faster for me to rewrite the code a bit than to make sense of it as it is – though fastidiousness has also played a role, I'm sure.
	I've stuck to the (Systems) Hungarian notation, mainly for consistency with the BtS code, but I also think that it has some merit in this software design. Since objects are usually passed around through integer ids (instead of pointers), it's typical to work with an object pointer and its id side by side, and then `eTeam` is shorter than `teamId`.
See also	003g, advc.fract: Fractional arithmetic 003e (private copy-constructors) makes the cast to CvCityAI safer to use. advc.pf: Refactoring of pathfinding code. advc.700, advc.250b: The R&F and SPaH code uses a bit of a different coding style. It took me a while to figure out which windmills (not) to fight. Eventually I settled on a middle ground and most of the codebase is consistent with that, but I haven't bothered to update the R&F and SPaH code because it isn't really subject to change by me and especially not by other modders.

Removed all trailing whitespaces in the C++ files, converted spaces used for indentation to tabs (also in the K-Mod Python scripts) and removed most of the spaces before closing parentheses.

Rationale	Shortly before releasing v0.96, I decided that it's better to make these changes once and for all instead of making them piecemeal along with functional changes. Irregular whitespaces are distracting to me when reviewing Git commits. For indentation, tabs were already used almost exclusively. It's especially important to stick to one way of indenting in Python. I haven't made further style changes in Python and none in XML

	because those files aren't frequently changed (by me).
Tbd.	Seach an replace (case-sensitive!): "if(" → "if (" "for(" → "for (" "while(" → "while (" "switch(" → "switch ("

CvDeal: Added a more convenient interface for iterating over trade items. When a CvDeal object and the two trading civs A and B are given, then the BtS interface with functions like `getFirstTrades` and `getSecondTrades` forces the caller to check whether A is either the "first" or "second" civ, resulting in redundant code to deal with both cases. The function names also give no indication who gives an item away and who receives it.

I've adapted all or almost all the client code for which it makes sense to the new interface.

AdvCiv	*BtS*

New class CvDLLLogger to encapsulate the "message log".	The "message (control) log" is used mainly for logging combat outcomes and random numbers (if "RandLog" is enabled in addition to "MessageLog"). And some misc. logging, e.g. in `CvPlayer::setTurnActive`:
	<pre>if (GC.getLogging())
	{
	    if (gDLL->getChtLvl() > 0)
	    {
	        TCHAR szOut[1024];
	        sprintf(szOut,
	          "Player %d Turn ON\n", getID());
	        gDLL->messageControlLog(szOut);
	    }
	}</pre>

Rationale	To make the logging code less distracting.
See also	There's a short guide about the various logs under 007.

Renamed the `getBugOption`... functions to "isEnabled" and "getValue" and put them in a namespace "BUGOption".

Merged a few non-functional changes from Roamty's Unofficial Patch 1.7 (link). Not marked in-line because the changes are too minor. He uploaded further changes here, but those are really just whitespace changes; nothing merged except for a bugfix (see "Roamty" under 001).

AdvCiv	*BtS*
"GC" is now a `CvGlobals const&`, and most CvGlobals member functions have `const` qualifiers.	"GC" (global context) is a reference to the singleton instance of CvGlobals. It doesn't have a `const` qualifier. Most of the CvGlobals member functions don't have `const` qualifiers either, although they don't change the state of CvGlobals.
`CvXMLLoadUtilitySet.cpp` redefines GC as a (non-`const`) `CvGlobals&` for convenient access.	The main class that changes (initializes) the state of CvGlobals is CvXMLLoadUtility.
Functions that provide references to other singleton objects, in particular getGame, getInitCore and getMap, still return non-`const` references. This works because CvGlobals only stores pointers to those sub-objects.	GC provides access to other frequently used singleton instances – CvGame, CvMap, CvInitCore – and to the non-synchronized PRNG and various pathfinders. Some of those instances are owned (allocated and deallocated) by

In the rare cases, when the state of CvGlobals needs to be changed by classes other than CvXMLLoadUtility, the instance is accessed through `CvGlobals::getInstance()`.	CvGlobals.
Rationale	Most classes have no business changing CvGlobals, so the `const` restriction on GC should serve as a warning. I'm not sure if any performance gains are possible; making GC `const` certainly can't hurt with regard to performance.
Removed functions from CvGlobals that had returned references to the info vectors. Instead, CvXMLLoadUtility accesses the vectors directly.	For each of the 100 or so "info" vectors that store data loaded from XML, there are three accessor functions: One that returns the size of the vector, one that returns an element at a given index and one that returns the whole vector by reference. The last function type is only called by CvXML LoadUtility, which is a `friend` of CvGlobals.
Rationale	Shouldn't expose those vectors to the entire game core when only one class needs to access them is already declared as a `friend`.
See also	advc.enum lets the preprocessor generate the remaining info accessor functions. Cavemen2Cosmos did the same thing (a little later than I): Git commit

Line endings: The DLL only uses line feeds (LF); I've run it through unix2dos (on devolution's suggestion). I think the original DLL also used LF exclusively, as did karadoc's code. XML line endings are mixed. Looks like BtS always used LFCR and so did I, but BUG uses LF. Well, at least line endings within individual files seem to be consistent. Haven't checked the Python code.

Turned K-Mod's CvFoundSettings and `AI_foundValue_bulk` (CvPlayerAI) into a new class CitySiteEvaluator.

Rationale	`AI_foundValue_bulk` was one of the most complex functions in the code base and it was located in the longest and most complex implementation file (`CvPlayerAI.cpp`). Moving it to a separate class has allowed me to store the data that gets computed prior to the city radius evaluation loop in member variables. This was a prerequisite for moving code into subroutines.
Tbd.	It's easy to see that the code still isn't well structured (for one thing, several functions return multiple values through reference parameters) and that the order of the evaluation steps is fairly arbitrary.
See also	031 deals with functional changes to the found value computation. When I moved the code, I also added log output; see 031c about that.

Converted most of the `CvArea*` function parameters to `CvArea const&`. Those changes aren't marked with comments. Repurposed the `getArea` functions (CvPlot, CvCity, CvUnit) to return a `CvArea&` instead of an `int` id. Same-area checks are now usually performed by functions `sameArea`, e.g. `CvCity::sameArea(CvCity const&)`, or `isArea(CvArea const&)`. The `area` functions returning a `CvArea*` (CvPlot, CvCity, CvUnit, CvSelectionGroup) aren't used much anymore. All functions mentioned above are inlined except `CvSelectionGroup::area`. Previously, most of them weren't inlined because CvPlot and CvArea objects had to be looked up in CvMap. Now CvPlot, CvCity and CvUnit each store a `CvArea*`, and CvPlot no longer stores an area id.

Rationale	constness is preferable and so are references and inlining. None of this is important on its own, but all three combined gave me enough impetus for this rather expansive refactoring change.
	I've kept the `area` functions mostly for compatibility when merging with other DLL mods. The change to the `getArea` functions will hurt compatibility, but there really shouldn't be three different area accessors I think and accessing them by id is easily

	the worst method.
See also	The `CvMap::getPlot` function introduced by advc.opt also returns a reference, whereas `CvMap::plot` returns a pointer.

Refactored the "plot unit functions" in CvGameCoreUtils, mainly through more descriptive variable names, more assertions.

Tbd.	Replace this with functors taken as template parameter? For better compile-time type safety.

Moved the `getCombatOdds` and `LFBgetCombatOdds` functions from `CvGameCoreUtils.cpp` into a separate file `CombatOdds.cpp`, split them up so that they can share some code with each other and so that the Advanced Combat Odds mod (ACO) can also reuse some of it. Moved some of the ACO code from `CvGameTextMgr.cpp` into `CombatOdds.cpp`, the rest into `ACOText.cpp`. Refactored all the odds-related code a bit, mostly the variable names (ACO had used some obscure one-letter abbreviations).

Rationale	Should make it more feasible for mod-mods to change the combat rules. Updating the odds calculations is a real obstacle to that, especially when it has to be done in five(!) different places.
See also	advc.test: Code for verifying calculated odds through simulations.

Deleted all copies of English game text that were standing in as placeholder translations in BUG, BULL, K-Mod game text files. This concerns mostly French translations. Until v0.99, AdvCiv game text for the BUG menu and the R&F option had also used copies of English text as placeholders for French, Italian and Spanish; those have also been deleted now.

See also	This Git commit has enabled K-Mod to fall back on English text when a translation is missing.
Rationale	Makes the game text files a bit easier to scroll through and makes it more apparent where translations are missing. The downside is that any kind person who wants to add translations will have to copy the opening and closing tags, which is arguably a bit more work than deleting a placeholder text. On the other hand, the missing tags make it easy to search the text folder for all missing translations (e.g. the closing English tag being followed by a linebreak and then the opening German tag implies that the French translation is missing).
Tbd.	The game text could generally be organized better – I guess; not sure how. `Civ4GameText_advc.xml` is too big, and it's generally difficult to identify redefines that might clash when merging another mod into AdvCiv. At the least, a uniform comment should be placed at every text element that redefines a BtS key.

Deleted the copy of `Civ4GameText_Events_BTS.xml` (half a megabyte) that was included in the BUG mod and replaced it with a file `EventText_CorrectedTranslations.xml` that includes only the few texts that the BUG mod had actually changed.

Rationale	As a rule, modders should be free to assume that a BtS text key that they wish to change isn't already included in the mod. Therefore, wholesale copying like that is bad.
Tbd.	Would be very nice to let the DLL check whether a text key gets loaded more than once. For GlobalDefines, this is already implemented through the `CHECK_FOR_REDEFINES` preprocessor flag (tagged with "advc.test"); harder to do for text I think.

003b	Misc. performance tweaks (tagged in the source code with "advc.opt")
See also	003d: Faster Quick Load

	[advc.inl](#): Function inlining [advc.pf](#): Pathfinder optimizations [advc.enum](#): More efficient data structures for data stored for every value of an enum type. [003s](#): Cache tile adjacency lists [003h](#): Cache `MaxVisibilityRange` [003m](#): Cache `CvTeam::isMinorCiv` and `getAtWarCount` [003o](#): Profiling code 003p: Improve performance of layer updates. [003v](#): Don't load unused XML data [045](#): Rival buildings hidden on main interface. [106i](#) clear Hall of Fame data from memory when a game is started or loaded. [004s](#) uses a more efficient data structure for player statistics. Brief chapter about [performance](#) in the main portion of the manual.
Tbd.	Test if [this](#) `flat_set` implementation by alberts2 is more efficient than `std::set` in the UWAI component (`UWAISet.h`). Not easy to do because his `flat_set` doesn't have quite the same interface as `std::set`. The implementation in [Boost 1.35](#) seems to have the same interface, but it can't be easily isolated from the rest of that library and I don't want to switch from Boost 1.32 to 1.35. If I do upgrade Boost, then I should follow billw2015's example (C2C [Git issue](#), [branch](#)) and upgrade to Boost 1.55.

I've tweaked various BtS functions that the Unit AI uses to quickly dismiss tiles when deciding where to move: `CvUnitAI::AI_plotValid`, `CvUnit::isEnemy`, `CvUnit::isPotentialEnemy` and related functions at other classes. The "potential enemy" functions check for imminent war plans, so I've moved them to AI classes. Some are used by the AI to avoid danger – I've kept the name "potentialEnemy" for those –, others are used for planning attacks – I've renamed those to "mayAttack". It's still a bit of a mess.

A few of my changes to calls of `isEnemy` are minor functional changes, bugfixes arguably, that only matter when a mod-mod makes additional units `AlwaysHostile` and when such a unit is near a friendly city or fort. I haven't marked those changes with any comment.

See also	Often, Unit AI code, especially BBAI code, checks whether a tile is in the same area as the unit. Change [030](#) (peaks and ice can separate areas) complicates some of those checks because submarines can now sometimes enter a different area. I've added functions `CvUnit::canEnterArea`, `CvArea::canBeEntered` and `CvUnitAI::AI_canEnterByLand` for dealing with that.

Revised access to GlobalDefines. Integer GlobalDefines are now accessed in four different ways: • Access through `CvGlobals::getDefineINT(char const*)`. This is still the default method for code that is clearly not going to affect the overall performance. In a few cases (e.g. in `MapGenerator.cpp`), I've at least moved `getDefineINT` calls out of loops and assigned their results to local consts instead. • Assign the result of `CvGlobals::getDefineINT(char const*)` to a local static variable. This is done when the value should only be used locally, i.e. when it's not really intended to be a "global" define.	Two different ways: • `CvGlobals::getDefineINT(char const*)`, which looks up the string argument in a `stdext::hash_map` (`FVariableSystem::m_mapVariableMap`) that contains all the GlobalDefines tag names.

• Enum value (`enum GlobalDefines`) in `CvGlobals.h` for fast access through `CvGlobals::getDefineINT(GlobalDefines)`. For values used in multiple places or that may well be used in additional places in the future. The mapping between XML element strings and enum values is handled by a macro. • Access through a dedicated function without parameters; e.g. `getEVENT_MESSAGE_TIME()`. For pre-AdvCiv functions that can't be easily removed because of `DllExport` or because of a high number of call locations. Most of these I did remove.	• For more than 50 frequently accessed tag names, there are individual variables cached at CvGlobals with a getter function, e.g. `m_iRIVER_ATTACK_MODIFIER` and `getRIVER_ATTACK_MODIFIER`.

Rationale	String retrieval from a map is fine in contexts where performance isn't a concern, but GlobalDefines are needed pretty commonly in contexts where it's at least not obvious that performance doesn't matter, and, then, adding a data member with a getter to CvGlobals is tedious and clutters that class up. The GlobalDefines enum solves that problem. The only reason I like to use local static variables sometimes is that it conveys that the GlobalDefine is only accessed in a single place. Caveat about local static variables: They're not updated if `CvGlobals::setDefineINT` is called. Another thing to be aware of: static local variables result in a branch instruction.
Credits	The local static idea came from Nightinggale. [CFC post](#)
See also	I posted about the GlobalDefines enum [here](#) in the C2C subforum, and a variant of that idea was later included in C2C ([Git commit](#)).
Tbd.	CvBugOptions could use a cache as well.

Added a parameter for default values (to be used when a tag isn't found in the hash map) and int-to-boolean conversion (`getDefineBOOL`) to the GlobalDefines getters in CvGlobals. This has rendered the `getDefine`... functions in `CvBugOptions.h` obsolete, so I commented those out.

Rationale	Treating integer defines as boolean is frequently needed and error-prone.

Faster functions for mapping CvMap coordinates to CvPlot objects: `CvMap::getPlot` returns a `CvPlot&` without performing any checks outside of assertions. Force-inlined. `CvMap::plotValidXY` returns a `CvPlot*`. Applies world-wrap, but doesn't check whether coordinates are -1. The compiler may or may not inline it. Throughout the game core code, `getPlot` is now used when coordinates are guaranteed to be valid. `plotValidXY` is used when the coordinates have been computed as an offset from valid coordinates. `plotSoren` is only used in a few locations where the coordinates may indeed represent no valid tile (and a few where I wasn't sure). `plot` is still used pretty widely, mainly in functions that receive coordinates through their call parameters.	There are two functions for this mapping, both force-inlined, both returning a `CvPlot*`: `CvMap::plot`: Applies world-wrap and checks if either coordinate equals -1. `CvMap::plotSoren`: Doesn't apply world-wrap, only checks for -1. CvPlot isn't generally used for data members; usually coordinates are used instead, and when those coordinates aren't supposed to represent any tile, they're set to -1 (e.g. `CvUnit::m_iReconX`).

Based on benchmark tests, I've kept `plotSoren` inlined and removed the (force-)inline keyword from `plot`. `getPlotByIndex` (returning a `CvPlot&`) mostly replaces `plotByIndex`. `getPlotByIndex` only asserts that the index is within the array's bounds.	For looking up a CvPlot by its index in `CvMap::m_pMapPlots`, `plotByIndex` is used. That function performs an out-of-bounds check and returns a `CvPlot*`.
Rationale	Speed and code clarity. `plotSoren` is a terrible function name. Now that fucntion is at least not used much anymore.
Tbd.	Functions shouldn't, for the most part, take coordinates as parameters; instead the should take a CvPlot reference if the tile is guaranteed to be valid and a CvPlot pointer otherwise. That should get rid of most of the remaining `plot` calls and superfluous `NULL` checks. Could then perhaps rename `plot` to "plotSafe".

003c	Added an assertion that checks if XML data loaded through `GC.getDefineINT` actually exists. Found just one (unimportant) error this way and corrected it. Had to change some calls that happened before XML was even loaded; no functional change.
	For values that CvGlobals caches, there is still no check if XML has been loaded. I've added an `isCachingDone` check to every getter function and found only one issue (`TechPrefs.py`; fixed). Then I removed the checks again because they were a bit unwieldy.

003d	Faster loading of savegames	
AdvCiv		*BtS*
When using Shift+F8 to Quick Load in fullscreen mode, the game exits to the main menu for a second before loading the Quick Save slot.		When loading any savegame while playing in fullscreen mode, the game hangs on "Initializing" (doing who-knows-what) for a period of time that appears to depend on the complexity of the game state. Can easily be 30 seconds. This delay does not occur when loading from the main menu (nor when exiting to the main menu).
Rationale	This (apparent) bug has been long known, though these three CFC threads are the only web sources I can find that mention it. (Upd.: Here's a more recent thread on Reddit; here a recent one on CFC. Upd.#2: Now I've posted about it.) It's possible that it only occurs on some systems. My fix only covers Quick Load. Since the "Load Game" menu is outside the SDK this is probably all I can do. Workarounds: a) manually exit to the main menu before loading; or b) play in windowed mode; or c) press Alt+Tab two times when the delay begins. This seems to finish the "Initializing" stage immediately. Could easily replace the "Initializing" text with a recommendation to minimize the game if loading is taking long. I've tested this and found it quite helpful because I keep forgetting about this. But the recommendation would also show up when generating a map, and this is misleading because Alt+Tab does not speed up the map script. The	

	replacement text also needs to be very short; no room to explain that it only applies when loading a savegame. Had to leave it at a regular loading screen hint (008d) that shows up randomly.
See also	009 is a prerequisite because it repairs BUG code for finding the `My Games` directory where savegames are located. 004m is a prerequisite because it makes sure that returning to the main menu doesn't affect the layers that are enabled after loading.

003e	Prevent accidental copying of objects	
AdvCiv		*BtS*
CvPlayer, CvTeam, CvGame, CvMap, CvEntity (base class of CvCity and CvUnit), CLinkList, CvInfoBase, CvScalableInfo, CvActionInfo (for the `get…Info` functions) and their subclasses are non-copyable, meaning that a line like `CvPlayerAI kPlayer = GET_PLAYER(ePlayer)` leads to a compile-time error.		Calls to the (macro) functions `GET_PLAYER`, `GET_TEAM`, `getGame` and `getMap` are extremely common in the code base, and new code has to call them all the time too. Such a line only leads to a runtime error, and one that is usually difficult to debug.
Rationale	Accidental copies are just something that comes with references as return values in C++, and I'm not going to make every class non-copyable that is returned as a reference somewhere, but securing these frequently used functions seems prudent.	
Credits	Git commit by Nightinggale (We the People mod). He does it only for `CvInfoBase`.	
See also	003k also makes a couple of classes non-copyable because it adds dynamically allocated memory to them. A public, implicitly-defined copy constructor for classes that free memory in the destructor violates the rule of three (but accidental copies are annoying even when they don't lead to use-after-free errors).	

advc.inl	Function inlining [the change id was "advc.003f" until AdvCiv 0.97]
I haven't exhaustively checked every header for functions that may benefit from inlining (though I did scour some of them), but I think I've taken care of the bulk of the functions that are very short, are at least somewhat frequently called and don't contain any conditionals. In most cases, I've merely moved the definitions into the header file. I've added the "inline" keyword only where the linker requires it for the one-definition rule. Note that MSVC03 (contrary to the C++ standard I think) never requires the "inline" keyword for function templates and member functions of templates, regardless of whether they're defined in-class or out of class; I guess they're implicitly treated as inline. I've added the MSVC "__inline" keyword almost exclusively to free functions in implementation files; it seems that the `Ob1` compiler option can't inline-expand such functions otherwise. I use "__forceinline" in only a couple of places where performance measurements have suggested that it's probably benefitial.	
I've also removed most of the (rather few) "inline" and some "__forceinline" keywords from Firaxis code.	
See also	Official documentation for the MSVC03 compiler: VS2003_CPP_en-us.pdf • "*The __inline keyword is equivalent to inline*" • "*__forceinline […] tells the compiler to go beyond the current inlining heuristics and to absolutely inline [...] except in cases in which inlining would be impossible.*"

	• "*/Ob1 expands only functions marked as inline or __inline or, in a C++ member function, defined within a class declaration.*" •"*/GL allows the compiler to perform optimizations with information on all modules in the program [i.e.] optimize the use of registers across function boundaries [and] inline a function in a module even when the function is defined in another module.*" advc.make: Use of compiler options by the AdvCiv game core DLL. "We the People" Git issue about function inlining. 003u partially solves problems with dependencies between abstract non-AI classes and AI classes derived from them. This has allowed me to inline additional functions without undesirable header inclusions.
Rationale	Based on measurements before and after moving groups of frequently used function definitions into header files and with and without whole-program optimization (`/GL` compiler option), I estimate that the lack (by and large) of inline expansion in the BtS DLL adds somewhere between 10% and 35% to AI turn times; a significant overhead. The use of `/GL` completely gets rid of this overhead in final-release builds; however, I don't use final-release builds during development (nor for profiling), so inline definitions still make sense for performance, and, in my opinion, have no drawbacks with respect to readability. I use "inline" and "__inline" side by side in order to differentiate between cases where an inline keyword is required and cases where I only want to nudge the compiler toward inlining. This is only my personal convention though; the two keywords are equivalent. Prior to AdvCiv 1.0 I had used inline keywords copiously, especially on wrapper functions, in order to reassure the reader that performance won't suffer (at all) or to point out functions that get very frequently called – as a warning against expanding those functions. I still think this latter rationale has some merit, but, on the bottom line, I've come to conclude that it's better to get rid of the inline clutter. In a test, the old code with all the inline keywords ran 5 permille faster than the newer code without them; since such measurements are affected by some noise, there may in fact be no difference in performance at all. That is, although the MSVC03 compiler is old and is by Microsoft, it does seem to have a solid, reliable logic for inlining. In my tests, force-inlining functions that branch at runtime and have multiple call locations has usually hurt performance. I'm guessing that this is mainly due to branch misprediction. Hence, there is generally no point in defining functions with branching in header files (such functions are also complex enough to affect readability).
Tbd.	As a matter of uniform style, all one-line const accessors should be defined in header files – but this is pretty unimportant.

AdvCiv	*BtS*
Removed the "INLINE" suffix from all function names and deleted the non-inline versions of those functions except for a few functions with const-correctness issues; those few functions I've renamed by appending the suffix "External" and then exported them under their original name through a module definition (`.def`) file.	The original developers have inlined only a few very frequently called functions. They've done so by defining them in-line and adding the `inline` keyword. No exported functions have been inlined – supposedly, in order to avoid inline-expansion at the call locations in the EXE; instead, separate non-exported functions with "INLINE" appended to their names (e.g. `CvGlobals::getGameINLINE`, `CvPlot::getX_INLINE`) were created for (all) DLL-internal calls, while the non-inline versions are only called externally. This convention has not been strictly

	followed, at least not in code added by the BtS expansion, and especially not by modders.
Rationale	To unclutter the DLL code and to make sure that only one version of each function is used in the DLL. (Now the `inline` versions are both faster and more readable than the exported versions, there is no reason for anyone to use the exported versions anymore.)
	I don't think the "inline" keyword is really crucial here. Inline-expansion of exported functions works just fine in the DLL without "inline", just so long as the function is defined in a header file. It might make a difference for inline-expansion at external call locations, but I doubt it. So I guess the Firaxis developers may have been correct that separate functions for internal and external use are needed when in line-expansion isn't ruled out by defining the function in an implementation file.
	That being said, so long as the source code of the EXE is unavailable, inline-expansion at external call locations isn't a problem. If it does become available (and that would seem like a remote possibility), then recompiling the EXE after any change to an exported function doesn't sound too problematic either.
	It would be kind of nice to add a …`External` version for all external functions and to export exclusively through the `.def` file. That would minimize the risk of breaking a DllExport, would make it easy to make minor changes to the signatures of exported functions and could avoid problems with inlining if the source of the EXE is ever released. However, this is, for starters, a big effort, and I also don't think it's going to be all the helpful on the inlining front: If the source of the EXE becomes available, one will want to use a [more recent compiler](#), which may use different [name mangling](#) and thereby break the `.def` file. Moreover, allowing the EXE to inline DLL calls would improve performance a bit.
Credits	I learned about `.def` files from posts by alberts2 and Anq in the Caveman2Cosmos forum ([thread](#)). Until then, I thought that it was impossible to rename functions that the EXE calls.
Config	The `.def` file is enabled through `Makefile.project`. Danny's/ Nightinggale's makefile had already supported `.def` files; only a linker argument had to be added.
See also	[CFC post](#) explaining how to add a `.def` file to a mod

Removed most of the `_USRDLL` preprocessor checks.	When disabled, the `_USRDLL` flag hides functions, preprocessor defines and enums that the EXE isn't supposed to access. That includes all exported inline functions.
Rationale	Hiding the "INLINE" functions was, as far as I can tell, pointless because they weren't exported. Hiding enums and defines was prudent at the time, but may not be necessary if the source of the EXE is released (see above) and is, in any case, not going to be crucial. And currently, to be clear, `_USRDLL` is always enabled – as it needs to be when compiling the DLL without the EXE.
See also	[Two](#) [posts](#) by EmperorFool about the `_USRDLL` flag.

CvPlot: Some frequently called functions have a (mandatory) `bDebug` parameter that allows ignoring the fog of war (for Debug mode), e.g. `CvPlot::isRevealed(TeamTypes,bool)`. I've kept the bodies of those functions in `CvPlot.cpp`, but overloaded them with functions that don't take a `bDebug` parameter and inlined the bodies of those new functions. (This way, the inline versions don't perform any conditional branching.) Then I've removed `bDebug=false` from all call locations so that all non-UI code uses the inline functions.	
See also	[advc.enum](#) simplifies many getter and setter functions, making them candidates for

	inlining.

003g	Utility functions for dealing with floating-point numbers (Still in place, but, as of AdvCiv 1.0 not used much anymore.)

The original game completely avoids floating-point math in all synchronized code, at least in the game core DLL. (I believe map scripts need to be synchronized, but they do use floating-point math.) The reason for this was, apparently, that floating-point operations are, in a sense, non-deterministic.

Much of the AdvCiv AI code, especially in the UWAI component, had initially used floating-point math, but, as of version 1.0, has been converted to fixed-point fractional math; only `std::log` is still used in a couple of places. K-Mod uses floating-point math for AI combat odds (via the `LFB_USECOMBATODDS XML` switch, which is enabled by default; BtS had used that code only for humans). Since K-Mod 1.46, floating-point math is also used for the AI evaluation of Cottages (Git commit).

So long as all players use copies of the same game core DLL, I don't think there can be a problem. The x87 (via the IEEE 754 standard), SSE and SSE2 instruction sets all guarantee the exact same results regardless of the specific CPU. I don't know if the MacOS version of Civ 4 is multiplayer-compatible with the PC version in the first place, but, since DLL mods run only on PCs, cross-platform multiplayer isn't a concern anyway.

One potential issue that I can see: A call to a DLL other than the game core DLL might change the floating-point precision or rounding mode, e.g. `d3d9.dll`, and a different version of that DLL might not make the same change under the exact same circumstances. That never seems to have occurred though (unsurprisingly I guess). I've added a test at game start that warns players when their floating point settings (apparently) differ, and, to my knowledge, no player has encountered that warning. (But, then, I've only heard from five or so multiplayer groups, and a test at startup can't catch problems caused by an FPU mode change in the middle of a game ...)

Tests prior to AdvCiv 1.0 (i.e. when floating-point math was heavily used) on a single machine with Microsoft's `_controlfp` function and different compiler flags suggested that inconsistent floating-point settings can indeed lead to unacceptably frequent OOS errors.

See also	My test at the start of a multiplayer game is based on the numbers in these two Stack Exchange questions: 1 2
	advc.fract: Class for fixed-point fractional math
	advc.make: The AdvCiv makefile enables SSE2 instructions (but I haven't verified that they're actually being generated).
	Posts by AIAndy arguing that floating-point math, fundamentally, isn't a problem.
	Post by Nightinggale arguing that it's difficult to be certain.
Tbd.	If it turns out that libraries mess with the floating-point settings, I could try to repair that through `_controlfp`: "At app startup time we call: _controlfp(_PC_24, _MCW_PC) _controlfp(_RC_NEAR, _MCW_RC) Also, every tick we assert that these fpu settings are still set: gpAssert((_controlfp(0, 0) & _MCW_PC) == _PC_24); gpAssert((_controlfp(0, 0) & _MCW_RC) == _RC_NEAR); There are some MS API functions that can change the fpu model on you so you need to manually enforce the fpu mode after those calls to ensure the fpu stays the same across machines. The assert is there to catch if anyone has buggered the fpu mode. FYI We have the compiler floating point model set to Fast /fp:fast (but its not a requirement)" source, similar advice, cf. MS library The `/fp:precise` compiler flag could also help.
See also	001n also deals with OOS issues

fract	Data type for fixed-point arithmetic: `ScaledNum`
See also/ Tbd.	[CFC thread](#) Comments that start with "tbd." in `ScaledNum.h` See [advc.enum](#) about the integration with `EnumMap`. [003g](#): Concerns about floating-point math
Rationale	To reduce rounding errors, for better readability and because a fractional power function is pretty indispensable for AI code in a game that revolves around exponential growth. Performance also improved a little (about 6% shorter AI turns) when I converted the UWAI code from `double` to `ScaledNum<2048>`; however, I made some non-functional changes along with that conversion that could explain at least part of the performance gain. There is still potential for optimization through intrinsics in the `mulDiv` function.

003h	Cache maximal visibility range
BtS goes through all improvements each time that the terrain or feature of a tile changes. That's a bit wasteful and easily amended.	
Credits	Adopted from the We The People mod ([Git commit](#) by devolution)

003i	Removed unused DLLExports and XML cache
AdvCiv	*BtS*
In the DLL, only functions that the EXE (either `Civ4BeyondSword.exe` or `Civ4BeyondSword_PitBoss.exe`) actually calls have the "DLLExport" keyword. I've removed the unnecessary DLLExports based on [Dependency Walker](#) output. Of course that tool can't detect unreachable code, so there may still be some unnecessary DLLExports.	Functions called by the EXE have to have the DLLExport keyword, otherwise the game crashes when the call happens. Beyond these functions, Firaxis programmers have added the keyword to several hundred functions that they perhaps thought they might want to call from the EXE; and, in some cases, DLLExport was obviously added indiscriminately.
I've removed DLLExport from structs and instead added it to the individual functions that the EXE calls except for cases where every (explicit) function was called by the EXE.	The DLL also declares some structs as DLLExport. This means (apparently) that all functions of the class, including implicit ones, can be called by the EXE.
For functions that do require DLLExport, if I have to add a parameter, I'm employing this pattern (in the header file):	Patch 3.19 removed a lot of unused DLLExports, but not nearly all of them.

```
DLLExport ReturnType functionName(Parameter1
param1) {
    return functionName(param1, defaultVal);
}
ReturnType functionName(Parameter1 param1,
Parameter2 param2);
```

Where `param2` is the new parameter and `defaultVal` its default value. In the source file, the new parameter is simply added to the function implementation.

Credits	Nightinggale; the pattern for overloading a DLLExport I've adopted from karadoc (he uses it on `CvSelectionGroup::canMoveOrAttackInto`).
See also	These [two](#) [threads](#) on CFC
Rationale	Makes it easier to change function signatures: If if it's not a DLLExport and not virtual, there is nothing to worry about. (Well, the DLL also passes some of the pathfinding functions to the EXE as pointers …) Also makes it a bit easier to guess what the EXE does when browsing the code.
Tbd.	Should perhaps make an internal copy of every exported function, append "External" to the name of the exported version by means of the module definition (`.def`) file (cf. [advc.inl](#)) and forward from the external version to the internal version. The external versions could be moved to the end of the class definitions (as I've already done it with most of the pure virtual functions, see [003u](#) – though the exported functions would have to retain their current visibility specifier). This way, one would never again have to worry about breaking a DLL export.

When the AdvCiv mod is loaded, it never attempts to read or write to the XML cache. The 2 times 19 `read`/`write(FDataStreamBase*)` functions in `CvInfos.h` are removed through a preprocessor flag. These functions are evidently only used for the XML cache.

The serialization functions used by the cache still cover all the relevant data. Well, probably – I can't verify that this is the case through tests. I've removed all code for compatibility with legacy cache formats.

Short version: I don't think the cache works at all in mods. Long version (partly guesswork):

When BtS is loaded, `crc.dat` in the XML cache is checked. (As for the cache location, see [this](#) post.) A checksum is computed from some of the XML files; it's not clear to me which ones and – if a mod is loaded – which version of the files. If the checksum matches the one in `crc.dat`, some 18 or 19 CvInfo objects are initialized from `CIV4...Infos.dat` files (binary format) in the cache, which is practically instantaneous. Otherwise, the CvInfo objects are initialized by parsing the respective XML files and `dat` files created from their serialized data and the new checksum. This takes about a second for the unmodded game, but can take a couple of seconds when a mod is loaded (especially if a debugger is attached).

The `DisableCaching` switch in the INI (`Beyond the Sword\CivilizationIV.ini`) prevents the cache read, but doesn't prevent the cache write. `DisableFileCaching` probably refers to the `catalogCiv4Beyond Sword....dat` files in the cache, but doesn't prevent them from being written. `DisableCaching` probably disables both the XML and the file cache from being read; difficult to verify.

	When loading a mod: Regardless of the INI settings, in my experience, the XML cache is never written. Whether the EXE attempts to read the `dat` files written during an earlier (unmodded) launch is unclear. This would result in erroneous behavior (probably an immediate crash). Clearing the cache is a very common recommendation on CFC and other websites for players who have trouble loading a mod. It's conceivable though that these problems have been resolved by one of the official patches, perhaps by simply disabling cache accesses when loading a mod. Another hypothesis: The XML cache has always been disabled for mods, and only the file cache (catalog files) has been causing crashes at launch. It's also not clear what the effect is of holding down Shift during launch. I don't see any cache files getting deleted; perhaps it merely suppresses the cache read.
Credits	Nightinggale; [CFC](#) (same thread as above)
See also	alberts2 has also removed the read/write functions from Caveman2Cosmos: [SVN revision](#) The DLL can disable INI settings as shown [here](#) by Nightinggale. However, the (non-XML) file cache probably can't be disabled that way because the EXE either doesn't call the DLL in between reading the INI and dealing with the file cache, or because CvDLLUtilityIFaceBase doesn't get instantiated in time. Also, the DLL can't find out the original INI settings, so all it can do is change them blindly without ever restoring the original settings. That's poor form because the changed settings get stored on disk, typically in the global INI file, affecting the behavior of all mods and the unmodified game. Large mods can take longer to load when they are launched for the first time. This is probably caused by caching in the HDD ([post](#) by Nightinggale); the XML cache is only a minor factor and the file cache seems to have no noticeable benefit. I've been trying to get a hold of the contents of some user's cache that will cause a crash at launch (don't really care with which mod). keldath posted some files [here](#), but I can't reproduce the problem and he can't rule out that his user profile was responsible for the crash. Related [post](#) in the Dawn of the Overlords thread As for backwards compatibility with old cache data: That would only matter if someone gets the cache to work for AdvCiv. And, then, there ought to be a better way, i.e. some way to disregard the whole cache on the first read after a version change.
Rationale	The slight speedup when launching would be worth something to me, but since it doesn't seem to work at all and might even lead to errors, it's clearly best to make sure that the cache is entirely disabled. Removing the read/write functions shaves 100 KB off the DLL file size.
Config/ Tbd.	The preprocessor flag is set in `CvInfo_Base.h`, and it's named `ENABLE_XML_FILE_CACHE`. Fixing the XML cache doesn't seem like an impossible task, and the `read`/`write` functions would be needed then. Perhaps those functions could also be used for other purposes.

	Regarding a rewrite of the XML cache code (I certainly won't do that): One would have to replace `cacheWrite` (declared in `CvDLLUtilityIFaceBase`), possibly `cacheRead`, and `CvCacheObject`, and write a new checksum test. String data in the EXE suggest that MD5 is used:
	`NOT Using XML cache, failed to find crc.dat file ... NOT Using XML cache, cur md5=%s, cached md5=%s ... Using XML cache`
	That part appears to be disabled (or broken) in the EXE.

003j	Unused functions
Through [Cppcheck,](#) I've identified about 200 DLL functions that are neither called internally nor from the EXE (see 003i above). Some of these had been in the code since Vanilla Civ 4, others had been added by the BtS expansion or a mod; some had, apparently never been used, others had been obsoleted by later changes. Many unused functions were simple accessors, mutators or wrappers that might still become useful someday and don't do any harm – these I've left alone. Other, more complex, functions had been entirely replaced by better code; these I've commented out or removed them through the preprocessor. If it wasn't clear if a function could still be useful (and not just as a starting point for new code), I've added a comment or an `FErrorMsg` to warn modders (such as myself) that these functions could be outdated. (If they're never called, runtime errors can't be noticed and corrected.)	
Tbd.	To identify unused functions that are exposed to Python, one would have to run `Cppcheck --enable=unusedFunction` on a code base that excludes the Python interface classes and look each hit up in a repository of all Vanilla, Warlords, BtS and AdvCiv Python files.
See also	[kmodx](#) – these bugfixes also seem to be the result of a code analysis tool.
Credits	I've deleted a bunch of CvInfo setter functions that were used only locally – without leaving comments in the code. Those setters were identified by MattOttawa (C2C; GitHub [pull request](#)).
Config	Cppcheck is quite easy to use for a project like this. One basically just has to load the `.sln` file in the GUI. External headers aren't needed; I guess Cppcheck just ignores unrecognized symbols. Providing the `std` and `boost` headers, specifically, isn't even recommended. A checkmark can be placed for Windows libraries in the GUI; I guess that's prudent. I've also selected `Win32-unicode` as the target platform. "All build configurations" should probably be unchecked because, otherwise, the analysis will take several hours. The `ART_INFO_DECL` macro in `CvArtFileMgr.h`, `ART_INFO_DEFN` in `CvArtFileMgr.cpp` and K-Mod's `trait_info` macro had originally aborted the Cppcheck analysis. They had used the `##` operator in questionable ways; easy to fix, no need for some special VS build configuration disabling problematic code.
	When reviewing the results, "suppress selected id" can be used to ignore a particular type of warning throughout the project. These suppressions get stored in the `.cppcheck` file, along with the target platform. I've committed my `.cppcheck` file to the Git repository.

003k	Workaround for adding data members to classes with exported constructor for which a certain size is mandated by the EXE: CvSelectionGroup(AI), CvSelectionGroupAI, CvReplayInfo, CvXMLLoadUtility and CvRandom. In the first case, there were already comments about the issue in `CvSelectionGroup.h`, but, for CvReplayInfo, I had to guess based on otherwise inexplicable exceptions in the destructor, and

<table>
<tr><td></td><td>

CvXMLLoadUtility had, to my knowledge, only caused issues with Wine under Unix (crash at startup due to corrupted data in CvGlobals; cf. advc.wine), though adding further data members might've become a problem on any system.

My understanding is that this can be an issue for any class whose constructor or destructor gets called by the EXE. (But I'm not too worried about virtual destructors – which the EXE can in theory call, when the constructor isn't exported.)

I've verified that other classes whose size mustn't change (or at least not by much) are CvFractal, CvDiploParameters, FVariable, CvPopupInfo, CvPopupReturn and CvTalkingHeadMessage. I've put static assertions in the header files. For CvDiploParameters and FVariable, the order of the members must also remain intact. Looks like the EXE is making raw copies of instances.

CvInitCore is a special case. It doesn't get allocated externally (or at least I don't see how it could be), but the EXE still depends on parts of the memory layout (not just the size) remaining unchanged. Perhaps a `memcopy` call in the EXE? Warnings added, and an assertions for the class size – even though the class size isn't decisive.

</td></tr>
</table>

Nested class (akin to Pimpl idiom) added to CvSelectionGroup, CvSelectionGroupAI, CvReplayInfo and CvXMLLoadUtility. Also added copy constructors (C++ rule of three) and static assertions that ensure that the object size stays as in BtS. For CvFractal, CvDiploParameters, CvPopupInfo, CvPopupReturn and CvTalkingHeadMessage, I've only added assertions.	All these classes have exported constructors (CvSelectionGroup only via CvSelectionGroupAI) or assignment operators and virtual or exported destructors, and the EXE calls some of those functions. In CvSelectionGroup, there was already a warning in the BtS code against adding data members.
Rationale	I've added the nested classes as a workaround for those classes to which I wanted to add data members. It's prudent to at least warn about the issue because the potential memory errors can go unnoticed for a long time and then become very difficult to debug. Not sure if it's perfectly safe to replace one `int` or pointer member with a pointer to an instance of the nested class, but storing the additional data in some completely unrelated place would be really awkward.
Tbd.	My pointer-to-nested-class approach doesn't allow for const correctness. The accepted answer (class template PrivatePtr) to this Stack Overflow question sounds like a promising remedy.
See also	The Visual Studio debug heap (disabled by default; see advc.make) can help discover memory corruption caused by exported constructors and destructors. Pinpointing the cause – not so much. E.g. after adding two booleans to CvXMLLoadUtility, I got a message "*Heap block at 036262B8 modified at 036262D0 past requested size of 10*" upon exiting from the main menu (probably originating from the CvXMLLoadUtility destructor) and a breakpoint somewhere in `ntdll.dll`. Without the debug heap, there was no overt adverse effect under Windows, but under Unix (cf. advc.wine), the mod crashed reliably at startup. On a related note, Visual Leak Detector is also fairly easy to integrate with Visual Studio. Only need to install it and include `<vld.h>` in some implementation file (I've put it in `CvMemoryManager.cpp`). That said, neither the VS debug heap nor VLD find `free` calls on uninitialized memory (well, that wouldn't be a leak either). Dr. Memory might be a better alternative, but it keeps crashing at startup (both versions 2.3.0-1 and 1.11.0-2) unless I use the `-no_track_allocs`, `-leaks_only`, `-no_count_leaks` and `-no_replace_malloc` options – which render the whole thing moot. Using the `/Zi`

	compiler option (and the non-parallel "Debug" build config) instead of `/Z7` may also be necessary (but hasn't helped me). Relevant documentation: [installation], [compiler flags], and the chapter "Running Dr. Memory" in the documentation included in the download. As command arguments in Visual Studio, the full path to the EXE in quotation marks needs to be used, followed by (not in quotation marks) the arguments for the EXE.

003l	Support added for ArtStyle-specific unit button graphics. This had mostly already been supported, except in the plot list (icons shown for units in a selected tile) and the Sevopedia.
Rationale	See [these] posts. Only relevant for mod-mods.
Tbd.	The list of units shown by Sevopedia when clicking on the category "Units" still shows the generic button graphic. Can be fixed (in `SevopediaMain.py`), but would take a bit of time. The "More Naval AI" mod may have some useful code fragement. ([Git commit])
Also added (untested) UI support for unit and building classes without a default unit type, i.e. classes that only exist for unique units.	
Rationale	Seems easy enough to do; only relevant for mod-mods.
Credits	CFC users Toffer90 and Inthegrave: Git commit [1] [2]

003m	Variables added to CvTeam for keeping track of minor civ status and at-war counts.
Rationale	Speeds up the frequently used functions `isMinorCiv` and `getAtWarCount`. Note: Minor civs were probably introduced for the Vanilla Earth1000AD scenario. The BtS version of that scenario doesn't use minor civs, and I don't think official or bundled content does. However, mod-mods might want to use them, so minor civs should arguably remain supported,
See also	[033] could have a performance problem without this.

003n	Make sure that AI attitude and memory of and toward Barbarians and minor civs are never used nor updated.
Rationale	The updates waste time and require the code that computes attitude to handle non-major civs gracefully. Code that uses the (memory) attitude of/toward non-majors is dubious; it's helpful for future testing to disallow this. Doing so has already revealed some potential issues, e.g. Barbarian cities assigning fewer defenders in "Always Peace" games.

003o	Changes to profiling code
See also	[advc.make]: Profiler settings in `Makefile.project`
Tbd.	Merge the internal profiler from MNAI (lfgr)/ C2C (Koshling): Git commit [1] [2] The DLL-internal profiler indeed seems unreliable. It looks like the time spent on collecting samples for a given function with `PROFILE_FUNC` adds to the total time of the callers of that function.
Moved memory management and tracking code from `CvGameCoreDLL.cpp` to a new file	

`CvMemoryManager.cpp` and moved (non-memory) profiling code from `CvGameCoreDLL.cpp` to a new file `FProfiler.cpp`.

Rationale	Seems cleaner this way. The profiling code was actually difficult to find. CvGameCoreDLL.cpp is special insofar that it's the implementation file for the precompiled header. Changing that implementation file causes the header to be rebuilt. That behavior might be desirable for the memory managment code, but I'd rather rebuild manually than place all sorts of unrelated code into `CvGameCoreDLL.cpp`.

Removed calls to the DLL-internal profiler from some very frequently called functions: `CvPlayer::canBuild`, `CvPlayer::canTrain`, `CvPlot::verifyUnitValidPlot`, `CvSelectionGroup::alwaysInvisible`, `CvUnitAI::AI_plotValid`, `CvPlayerAI::AI_unitValue`, `CvUnit::canMoveInto` and various pathfinding functions.

And removed some least-concern functions just to make the profiler log easier to read.

Also commented out some calls that I had added myself – as a reminder that those functions are called very frequently: `CvPlot::isTradeNetworkConnected`, `CvPlot::isTradeNetwork`, `CvPlot::getPlotGroup`, `CvCity::canTrain`

Rationale	As a rule, I'd posit that any function called more than half a million times during one late-game turn shouldn't be routinely profiled. It seems that at least part of the profiling overhead of a function gets added to the times measured for its callers, so profiling very frequently called functions may lead to misleading results. Also don't want to make profile builds needlessly slow. It's better to use an external profiler (e.g. Very Sleepy) for identifying code that gets executed extremely often.
Tbd.	I've done what I could (mostly through inlining; see advc.inl) to speed the "hot" functions up. They still take up a considerable portion of the overall runtime. Will have to call them less frequently – by making the pathfinding code more efficient or by relying less on pathfinding.

AdvCiv	*WtP*

Profiler for counting CPU cycles based on the Time Stamp Counter register adopted from the "We the People" mod. For profiling functions that are fast but get called very frequently.

Config	`/DUSE_TSC_PROFILER` in `Makefile.project`. The target configuration needs to be "Profile" in addition.
Credits	Code by Nightinggale. Adopted from this Git commit.
Tbd.	Crash in the EXE upon exit (when TSC profiler enabled). Don't know if that also happens with Nightinggale's code; could be that the singleton pattern I implemented for the TSCProfiler class is causing this somehow. `CvGlobals::uninit` terminates correctly; the crash happens before the TSCProfiler instance is destroyed. Not a big problem; without a debug build, the crash isn't even noticeable.

Added a compiler flag `PROFILE_AI_AUTO_PLAY` to disable code fragments that may randomly skew results when profiling on AI Auto Play. So far used only to prevent the active player from receiving a map in trade.

003p	Improve performance of `CvPlayer::getGlobeLayerColors`
AdvCiv	*BtS*
The help texts for the indicators shown by the Resource layer are cached and only updated when the active player obtains a new technology,	The indicator help texts are (unfortunately) not computed when the mouse hovers over an indicator, but already (for all revealed tiles on the

a new national wonder, a new religion or when anyone completes a wonder of the world.	map) when a layer is enabled. Setting the `GlobeLayer_DIRTY_BIT` causes the currently active layer (if any) to be updated: indicator positions, colors and text. That bit is set when the active player discovers a technology, but also when a tile becomes revealed or when the owner of a tile changes.
	In BtS/K-Mod, the impact on performance seems to be negligible.
See also/ Rationale	[004w](#) increases the complexity of the resource text computation, not that much really, but enough to cause a noticeable delay (200 ms, say) whenever the Resource layer is updated. The cache practically eliminates that delay. The cache updates match the information displayed by 004w. (Though I'm not sure that the updates guarantee that the text is never out of date.)
Tbd.	The updates still seem a bit wasteful to me. Should check which layer is active before setting the dirty-bit. For example, I don't think the unit layer needs to be updated when the owner of a tile changes.

003q	Duplicate code in CvPlayer initialization moved into subroutines	
AdvCiv		*BBAI*
Most of the code shared by `CvPlayer::init`, `CvPlayer::initInGame` and the Change Player component is inside subroutines (not duplicated).		BBAI has added `CvPlayer::initInGame` in order to fix errors that had occurred when a player was added dynamically, i.e. as a colonial vassal (or through a mod like Barbarian Civ). Comment by jdog5000: "*copy of CvPlayer::init but with modifications for use in the middle of a game*"
Also moved war declarations of non-major civs against the new player from `CvTeam::init` to `CvPlayer::initInGame` so that the player is properly initialized when the war declarations happen.		That's about 100 lines of duplicate code.
		Then, for the Change Player mod component, parts of the `init` code are duplicated two more times.
Rationale	Some shoddy early work by jdog. Would've made adding new trait abilities a pain.	
See also	[kekm.24](#): The commit message mentions errors that occur when placing a new player in the slot of a previously defeated player.	
	[CFC post](#) reporting a bug indirectly caused by the BBAI initialization code (among other causes).	

003r	Deferred UI updates: Added code akin to `deferCall` (`BugUtil.py`) to CvGame.
See also	Needed for [004j](#), [085](#) and [001w](#).

003s	Macros for list iteration
See also	See the end of [003u](#) about the FLTA (FFreeListTrashArray) class.
	[CFC thread](#) where I've posted about some of my macros.

<table>
<tr><td colspan="2">C2C wraps the BtS list data structures into Boost iterators ([Git commit](#)) and then wraps those iterators into Boost macros with a functor algebra for filtering ([Git commit](#)). Powerful, but also a bit difficult to get used to and, more importantly, too much work to adopt, at least now that I've already implemented my own solutions; may even require a Boost upgrade. See also the (older, but still valid) comment about CLinkList traversal in C2C at the end of 003s.</td></tr>
<tr><td>AdvCiv</td><td>BtS</td></tr>
<tr><td>

"FOR_EACH_..." macros that expand to the BtS-style loops:

`FOR_EACH_CITY_VAR(pCity, kOwner)`

The iterator variable declared by the macro gets a unique name based on the line number (`__LINE__` macro). For a `const` element pointer, or an AI element pointer, there are macros `FOR_EACH_CITY`, `FOR_EACH_CITYAI`, `FOR_EACH_CITYAI_VAR`. (`FOR_EACH_CITY` with a non-AI const pointer is supposed to be the standard macro and the one most commonly used.) Same for CvUnit, CvSelectionGroup, CvArea and CvDeal.

I've replaced BtS-style loops with my macros everywhere in the game core code base.

A hints file (`cpp.hint`) is used to get Visual Studio to recognize the scope of the macros. Auto-completion (Ctrl+Space) still doesn't (reliably) suggest the name of the element variable (e.g. `pCity`), at least not in VS2010.

</td><td>

Loops over FLTA elements have the following form (example):

```
int iLoop;
for (CvCity* pCity = kOwner.firstCity(&iLoop);
pCity != NULL; pCity =kOwner.nextCity(&iLoop))
```

The `iLoop` variable should really be named "iIter" (which is the name of the formal parameter in `firstCity` and `nextCity`). It stores the FLTA-internal position of the iteration, and since the FLTA is a "`non-packed array`" (Firaxis comment), `iLoop` is not equivalent to a loop counter.

</td></tr>
<tr><td>See also</td><td>

Comments in `FreeListTraversal.h`

[advc.enum](#) defines a `FOR_EACH_ENUM` macro.

</td></tr>
<tr><td>Rationale</td><td>

The BtS loops are very clunky and the `iLoop` variable is a bit dangerous as it can be misinterpreted as a loop counter (I've been there once). A proper iterator like

```
for (CityIter it = kOwner.firstCity(); it != kOwner.lastCity(); ++it) {
        CvCity const& kCity = *it;
```

might be more idiomatic, but would've been more work to implement and the macros are rather more convenient to use. (Although I'd prefer to work with references.) Also, using iterators everywhere would lead to a lot of nested iterator loops with awkwardly-named iterator variables (or bugs resulting from name clashes).

The auto-complete issue is only a little bit annoying and may be a nonissue with more recent versions of VS.

</td></tr>
<tr><td>

When iterating through CLinkList objects, `const` CLLNode pointers (and for IDInfo nodes also `const` CvUnit and CvCity pointers) are used when possible. When it was obvious that a loop doesn't modify the current node through some side-effect, `while` loops have been replaced with `for` loops.

Typical form:

`for (CLLNode<TradeData> const* pNode =`

</td><td>

CLinkLists are used, primarily, for storing the units in a tile or AI group, the cities or units selected by a human player and the trade items in a deal. A typical loop looked like this:

```
CLLNode<IDInfo>* pNode =
kPlot.headUnitNode();
while (pNode != NULL)
{
    CvUnit* pUnit = ::getUnit(pNode->m_data);
    pNode = kPlot.nextUnitNode(pNode);
    // …
```

</td></tr>
</table>

```
pFirstList->head(); pNode != NULL; pNode =
pFirstList->next(pNode))
{
    TradeData data = pNode->m_data;
    // …
}
```

For units (cities similar):

```
for (CLLNode<IDInfo> const* pNode =
kPlot.headUnitNode(); pNode != NULL; pNode =
kPlot.nextUnitNode(pNode))
{
    CvUnit const& kUnit = *::getUnit(
pNode->m_data);
    // …
}
```

`while` loops like the one on the right are still used when there's a possibility that a node could be removed in the body of the loop.

Specifically for traversing the unit lists `CvPlot::m_units` and `CvSelectionGroup:: m_units`, a macro `FOR_EACH_UNIT_IN` (with AI/ non-AI and `const`/ non-`const` variants) have been added. CvPlot and CvSelectionGroup have the same interface for accessing their `m_units` member; therefore the same macro names can be used for both.

For traversing the trade items in a deal, a macro `FOR_EACH_TRADE_ITEM` (again, with variants) has been added.

The macros that iterate over `const` units and nodes, come with an assertion that checks after each iteration whether the list length has changed. I've disabled those assertions after doing some tests. It seems pretty unlikely that accidental changes during traversal will go unnoticed even without the assertion.

```
}
```

Rationale	The `while` loops are error-prone; in particular, it's easy to mix up the order of the `getUnit` and `nextUnitNode` calls or to insert a statement in between them that causes the `nextUnitNode` call to be skipped. Such errors can be difficult to debug. The `for` loops are also dangerous; deleting `pNode` in the body will cause a crash. `const` qualifiers make these loops a bit safer. That said, the current node could still be deleted as a side-effect. For example, telling a unit to join a different selection group while iterating over its current selection group will delete the unit from the current CLinkList., leaving `pNode` as a dangling pointer. Killing a unit will immediately remove it from the unit list of its current tile; mustn't do that in a `for` loop over the units in that tile. `const` qualifiers on the CvUnit pointers should mostly prevent such mishaps.
	`while` loops aren't entirely safe in that regard either. In the example on the upper right, it's OK to kill `pUnit`, but, once `nextUnitNode` has been called, `pNode` mustn't be deleted, and the unit contained in that node (which could be any unit except `pUnit`) mustn't be killed.
	It's preferable to wrap these details into macros, but it depends on the type parameter of the CLinkList what the macro needs to do exactly, and some frequently used lists

	are encapsulated by other classes and therefore not directly accessible to macros. The macros I've written cover a large portion of the CLinkList traversals in the codebase.
	Nota bene: Whether a unit or city in a CLinkList can be `NULL` before the first iteration depends on the circumstances. AI groups and plot lists should be updated immediately when a unit is killed – I think (that should be assumed). I'm even less certain about human selection lists (handled by the EXE).
Tbd.	Should arguably add macros for dealing with `CvDLLInterfaceIFaceBase::nextSelectionListNode` (used mainly in `CvDLLWidgetData.cpp`) and `CvPlotGroup::nextPlotsNode`.
See also	CFC thread about the dangers of CLinkList loops. Also suggests that dangling node pointers won't necessarily lead to visible errors and that, for better or worse, such errors could be more likely to result in crashes under Windows 10 due to more aggressive memory management.
	billw2015 has written iterator classes for the Caveman2Cosmos mod (CFC post; GitHub links 1 2 3 4 5 6).
	C2C also has some special assertions for identifying indirect changes to CLinkList nodes during traversal.

Each CvPlot stores a list of its up to 8 adjacent plots. The lists aren't guaranteed to be in a particular order except that orthogonal neighbors always have an even index in the adjacency list and diagonal neighbors an odd index. An adjacency list can be traversed through the `FOR_EACH_ADJ_PLOT` macro. Example:

```
bool CvPlot::isAdjacentToLand() const
{
    FOR_EACH_ADJ_PLOT(*this)
    {
        if (!pAdj->isWater())
            return true;
    }
    return false;
}
```

Apart from `const`/ non-`const` versions, the macro has variants that skip over diagonal or orthogonal neighbors (`FOR_EACH_ORTH_ADJ_PLOT`, `FOR_EACH_DIAG_ADJ_PLOT`).

Adjacent plots get computed on the fly by going through all 8 directions and calling the `plotDirection` function (CvGameCoreUtils; moved to CvMap in AdvCiv) for each. Example (already edited a bit for readability):

```
bool CvPlot::isAdjacentToLand() const
{
    for (int i = 0; i < NUM_DIRECTION_TYPES; i++)
    {
        CvPlot* pAdjacentPlot = plotDirection(
            getX(),getY(), (DirectionTypes)i);
        if (pAdjacentPlot != NULL)
        {
            if (!pAdjacentPlot->isWater())
                return true;
        }
    }
    return false;
}
```

| *Rationale/ See also* | Even with the `FOR_EACH_ENUM` macro (see advc.enum), it's obviously clunky to get the neighbors via the `DirectionTypes`. The SquareIterator (advc.plotr) isn't as efficient as the BtS loops (not for range 1, that is). The BtS loops aren't all that efficient either. Even after some optimization on my part, `plotDirection` requires several conditionals in order to deal with the map's world-wrap settings (and, as a result, can't be efficiently inlined) and one multiplication and one (modulo) division. The adjacency lists only increase the size of CvPlot by only 4 byte. Adjacency lists constantly need to be traversed during pathfinding. Thanks to K-Mod, all the pathfinding of AI units happens within the DLL and can therefore take advantage of the adjacency lists. I think the speed-up of AI turns I got from the adjacency lists was around 10%. That said, AdvCiv performs (far) more pathfinding than K-Mod does (see 104b). |

003t	More efficient storage of XML data (CvInfo classes)
AdvCiv	*BtS*

Most of tag pair lists loaded from XML are stored as a single `NULL` pointer if empty. I've implemented this in the `CvXMLLoadUtility::SetVariableListTagPair` functions (for various data types).
 	
In most cases, the functions for accessing the CvInfo data were already performing a `NULL` pointer check. However, for integer data, -1 was returned in the `NULL` case. Normally, the proper default value is 0 though, so I've had to make changes in a few dozen accessor functions.
	XML data with a map/ dictionary structure is given as lists of pairs of elements. For example
	`<FlankingStrikes>`
	`<FlankingStrike>`
	`<FlankingStrikeUnitClass>UNITCLASS_CATAPULT</FlankingStrikeUnitClass>`
	`<iFlankingStrength>100</iFlankingStrength>`
	`</FlankingStrike>`
	`<FlankingStrike>`
	`<FlankingStrikeUnitClass>UNITCLASS_TREBUCHET</FlankingStrikeUnitClass>`
	`<iFlankingStrength>100</iFlankingStrength>`
	`</FlankingStrike>`
	`</FlankingStrikes>`
	in `Civ4UnitInfos.xml`. The CvInfo classes store such data as arrays; e.g. for every unit an array with one integer entry for each unit class to store the flanking strike ability. If the list of pairs is empty (e.g. no flanking strike ability), all entries are 0.

See also	This technique has been **superseded** by enum map classes and [glue](#) that makes them easy to use for XML loading, however, it's still used in some places, so I'm leaving the documentation in place until all of the code has been converted.
Rationale	To store the data more compactly. In total, I estimate that only a few MB (5 maybe) will be saved, but, especially for CvUnitInfo, I'm hoping that saving a few KB per object will improve the performance of the CPU cache.
Credits	Inspired by Caveman2Cosmos (specifically [this](#) thread).

Added `isAny...` functions for some of the tag-pair-list arrays loaded from XML. The `isAny...` functions test if an array is allocated. Went through the call locations of the respective array accessor functions and inserted `isAny...` checks to avoid costly loops. In particular the checks avoid a loop over …
 	
• `BuildingClassTypes` in `CvCity::canConstruct`
• `BonusTypes` in the `CvCity::getProductionModifier` functions
• `BonusTypes` in the `CvCity::CvCity::getAdditional…ByBuilding` functions
• `BuildingClassTypes` in `CvCityAI::AI_bestBuildingThreshold`
• `BuildingClassTypes, ImprovementTypes` in `CvCityAI::AI_buildingValue`
• `PromotionTypes` in `CvPlayerAI::AI_unitValue`
 	
(Most of these loops are now handled by enum maps instead; see [below](#).)
	For example, this loop in `CvCityAI::AI_buildingValue`
	
	`for (int i = 0;`
	`   i < GC.getNumBuildingClassInfos(); i++)`
	`{`
	`   if (kBuilding.`
	`      getBuildingHappinessChanges(i) != 0)`
	`     iValue += kBuilding.`
	`        getBuildingHappinessChanges(i) * …`
	
	goes through all ca. 125 building classes to check if any of them has the building-happiness-from-other-building ability – which is actually unused.

Rationale	Not a great improvement overall, but could be helpful for future AI code.
See also	"More Naval AI" uses a separate class CvInfoCache for such optimizations (including the one in `AI_unitValue` listed above). [Git commit](#)
Store lists of prerequisites as vectors.	Arrays of a size set in GlobalDefines.xml; e.g.

The list length limits from `GlobalDefines.xml` are still checked through assertions (because Python code may rely on those limits).	`NUM_UNIT_AND_TECH_PREREQS`.
Rationale	Avoids unnecessary iterations in loops over all prereqs and, more importantly, makes it unnecessary to check for non-requirements (`NO_...`), which also improves readability and makes loops over prereqs less error-prone.

AdvCiv	*BtS*
Removed the `<ForceBuildings>` XML tag for `Civ4UnitInfos.xml`.	That tag is no longer used since the BtS expansion. Unlike the `<Buildings>` tag, `<ForceBuildings>` causes any preconditions that could prevent the unit from constructing the building to be ignored. Warlords used it only for the Academy. I not sure why the Academy may have required this special treatment. Perhaps buildings without any tech requirement used to be impossible to construct.
Credits	The Caveman2Cosmos modders noticed that the tag is wasteful and unnecessary.
Rationale	With the tag pair list changes above, the unused tag was actually no longer wasteful, but I can't think of a reasonable way to ever use this tag for anything.

AdvCiv	*BtS*
Use enum maps to store nested data loaded from XML, along with macros that make all the necessary declarations in the `Cv_...Info.h` files (also enabling fast iteration over non-default values) and CvXMLLoadUtility functions that insert the XML data into the maps. The choice between the available map types needs to be made at compile time – by calling the appropriate `DEF_...` macro in the `Cv_...Info` header.	The nested data consists mostly of lists of pairs that map some element of play (represented by an enum type in the DLL) to an integer value or to a tuple of yield or commerce values or modifiers. The DLL stores that data in two- and three-dimensional arrays, filling in a default value (normally 0) for all enumerators not listed in XML. (See above for an example.)
See also	advc.enum: The available enum map classes (which are also widely used outside the CvInfo class).
	The "We the People" mod uses a datastructure specialized for dealing with three-dimensional XML data. CFC thread (By encoding yield and commerce tuples in primitive data types, I'm mostly dealing with mappings from keys to primitive values, but the dozens of yield types in WtP rule that approach out.)
Rationale	I don't think much speed is being gained here over the my previous makeshift solution of collapsing empty arrays – which was already not a very significant improvement. The greater benefit is that the `DEF_...` macros unclutter the CvInfo classes and make it easier to add new two-dimensional XML data without worrying about performance.
	Letting the XML loading code choose an enum map type at runtime based on the number of non-default values would have added overhead for virtual function calls – which might well have neutralized any performance gains.
Tbd.	So far, I've converted only CvBuildingInfo from raw arrays to enum maps.

advc.tag	Non-nested XML tags as enum values
Rationale	Make it easier to implement XML schema changes on the DLL side and – later maybe – to reduce code duplication in the `read(CvXMLLoadUtility*)` functions of classes

	(currently) derived from CvInfoBase. I'm not sure if this is really a worthwhile approach. Anyway, it's optional; XML element tags can still be added as in BtS.
See also	[003t](#) defines macros for loading two-dimensional (i.e. nested) from XML more easily and efficiently. That technique is very different from the one described here, and I don't think they can be reconciled. Another reason, perhaps, not to refactor the loading code for the non-nested tags.

AdvCiv	*BtS*
Added an abstract class CvXMLInfo derived from CvInfoBase. To make a class that is derived from CvXMLInfo load an element tag from XML, it's enough to add the element name to an enumeration and to a virtual function named `addElements`, both in the header file. See CvImprovementInfo for an example. CvHotkeyInfo is derived from CvXMLInfo, i.e. all classes derived from CvHotkeyInfo are also derived from CvXMLInfo. However, so far CvBuildingInfo is the only CvHotkeyInfo class that I've added an `addElements` function to. Elements added in this way can be accessed through a generic `get` function; e.g. `kImprovement.get(CvImprovementInfo:: DefenseModifier)` if `kImprovement` is a reference to a CvImprovementInfo object. Only integer and boolean elements are supported so far.	All classes for loading and storing XML data are derived from CvInfoBase (except for XML data about artwork, which has its own class hierarchy rooted at CvAssetInfoBase). To load an additional element from XML, two additions need to be made in the header file: A public accessor function and a private data member. In the implementation file, the data member needs to be initialized by the constructor (not strictly necessary if the element is mandatory), the accessor function needs to be implemented (unless an inline definition is used) and, in `read(CvXMLLoadUtility*)`, a few lines of code need to be added that do the actual loading through CvXMLLoadUtility. If the XML file is cached by BtS (many of the frequently changed XML files are), then any new element should also be added to the `read`/`write( FDataStreamBase*)` functions. Example for an element access: `kImprovement.getDefenseModifier()` XML elements can contain integers (ids and scalar values), booleans, floating point numbers, strings and mappings, typically mapping integers or pairs of integers to other integers (e.g. `TerrainMakesValid`, `RouteYieldChanges` in CvImprovementInfo).

| *Rationale* | Ideally, to add a new XML element tag, the name of the tag would have to be specified only once in the DLL. This is possible through macro definitions and that works pretty well for GlobalDefines (see [advc.opt](#)) but requires too much error-prone code to be added to each info class in the case of non-global XML elements. Still, having to make changes in 2 places is better than the 4-7 places in BtS.

As for the new "CvXMLInfo" class: It represents XML element tags as enum values. That's probably not clear from the class name; don't want a long class name though because derived classes need to refer to some base members explicitly. At least one class (CvActionInfo) that doesn't get loaded (directly) from XML is derived from CvInfoBase, so one could argue that representing *only* XML data is a specialty of CvXMLInfo.

I've tried implementing the enum handling directly at CvInfoBase, but that resulted in a crash in the EXE. My guess is that there is some class derived from CvInfoBase (probably a small one) whose size mustn't change (cf. [003k](#)). |

See also	The `HealthPercent` tag added to CvImprovementInfo for [901](#) serves as an example.
	[003i](#) disables the XML cache, but I'd still like to keep the code for storing XML data on disk up to date. Through the CvXMLInfo class, this doesn't have to be done manually for every new XML element.
	[003t](#) allows cached GlobalDefines to be accessed through enum values.
	[xmldefault](#) introduces a new system for default values, but I've only implemented that for `Civ4LeaderHeadInfos.xml` so far
	Nightinggale has written a [Perl script](#) that generates enum values from the type elements in XML. By parsing the schema files, essentially all code in the CvInfo classes could be generated by such a script. (Without an external code generator, one could only end up with something inefficient like the `CvGlobals::getDefineINT(char const*)` function.)

Only rudimentary support for exposing XML elements to Python; a macro call needs to be added for every element to be exposed, and a definition pointing to the function declared by the macro needs to be added to one of the CyInfoPythonInterface files.	Most of the CvInfo accessor functions are exposed to Python. The CyInfoPythonInterface files contain pointers to the accessor functions.

See also	Comment above the macro `PY_GET_ELEMENT` in `CvInfo_Base.h`.
Rationale	I haven't found a way to expose my generic `get` functions to Python. That's a pity because it would be nice to make new XML elements available to Python scripts, and doing so manually is tedious.
	When converting BtS elements to CvXMLInfo, all Python exports should be kept intact. Exposing the generic `get` functions wouldn't help here; but the macro solution also isn't great as it requires a macro call for each tag exposed.
Tbd.	To allow elements to be added as enum values to a CvInfo class not derived from CvHotkeyInfo, its base class will have to be changed from CvInfoBase to CvXMLInfo and all explicit calls to base class functions will have to be updated to call the new base class instead. Also, even if derived from CvHotkeyInfo, an enum for each data type needs to be added to the header file and the `addElements` function needs to be overridden.

xmldefault	Default values for optional XML elements set through special `<...Info>` element
AdvCiv	*BtS*
When an `<...Info>` element of type `..._DEFAULTS` is present in an `Civ4...Infos.xml` file, then all subsequent `<...Info>` elements use the values of the `DEFAULTS` element as default values for optional elements. On the DLL side, this behavior is only implemented for `Civ4LeaderHead Infos.xml` so far. All LeaderHead data that is the same for all or almost all leaders is optional and has a default value set through `LEADER_DEFAULTS`. I've deleted all optional data that equaled the default value, reducing the file size of	Subelements of `<...Info>` are optional when the schema file sets `minOccurs="0"` for them. XML schema definitions – in principle – support default values, however, BtS (specifically the EXE) only uses the schema file for syntax validation. Loading XML data and setting defaults is up to the DLL, which has no access to the schema parsed by the EXE. Through the `Cv...Info` constructors, the DLL sets default values of 0, `false`, an empty string or (for enum types) -1 for all elements (regardless of whether they're optional in the schema). For `Civ4LeaderHead Infos.xml`, the BtS schema makes all data

`Civ4LeaderHeadInfos.xml` by more than 50%. LeaderHead XML files without a DEFAULTS element can still be loaded, so XML mod-mods shouldn't have to change anything.	mandatory although many elements are the same for all leaders, in particular the contact delay values.
Modular loading (not used by AdvCiv): The defaults set in `Civ4LeaderHeadInfos.xml` also apply to any LeaderHeadInfos loaded from a module. Modules should not define their own `LEADER_DEFAULTS`. For behavior similar to MRGENIE's "TrueModular XMLCOPY" mod, I've left a few lines commented out in `CvXMLLoadUtility::SetGlobalClassInfo`. Not really useful for LeaderHeadInfos I think.	In BtS, modules can only contain full definitions of `<...Info>` elements. Those elements are then added to those loaded from the main module or, if an element of the same type had already been loaded, the old element gets replaced. The XMLCOPY mod (included e.g. in Rise of Mankind) allows modules to alter individual subelements of a previously loaded `<...Info>` element instead of replacing the whole element.

Rationale	Mainly to make the behavior shared by all leaders easier to adjust, i.e. through a change in one place as opposed to 52. Uncluttering `Civ4LeaderHeadInfos.xml` is a nice bonus. An alternative (more laborious) approach would have been to parse the schema file in the DLL and get default values from there.
See also	006b warns about all optional elements not found by the DLL unless a default value is explicitly set in the `GetChildXmlValByName` call (overriding the initial value set by the `Cv...Info` constructor). If the `DEFAULTS` were kept around past XML loading, they could be used for 004q (see *Tbd.* there). advc.rh also deals with modular loading.
Tbd.	Add `DEFAULTS` for other large `Civ4...Infos.xml` files, e.g. UnitInfos. This will require (non-trivial) DLL changes akin to those I've made in `CvInfos_Civilization.xml`.

003u	Changes to the design of the AI classes
AdvCiv	*BtS*
Same basic design. The AI found value computation has been split off from CvPlayerAI into a component class CitySiteEvaluator.	The AI code is split across the classes CvTeamAI, CvPlayerAI, CvCityAI, CvSelectionGroupAI, CvUnitAI and CvGameAI. Those classes are derived from non-AI classes representing the teams, players, cities, unit stacks, units and overarching game logic. E.g. CvUnitAI is derived from CvUnit.
Rationale	I suppose the Firaxis programmer just wanted the AI code in separate classes and still be able to directly call non-AI member functions. That may not be the best possible design, but it's fair enough. Not all e.g. player-related AI code should've been placed in a single CvPlayerAI class, but splitting parts off through object composition is no problem.
Tbd.	I don't think polymorphism was the goal of using inheritance for the AI classes, but polymorphism could be useful for structuring the AI code. Separate classes for the Barbarians could make sense as the Barbarian AI behavior is in many ways completely different from the regular AI behavior. CvUnitAI could be split into

	LandUnitAI, SeaUnitAI and AirUnitAI – the three don't have much in common.
See also	billw2015 has endorsed composition as his means of choice for breaking up the large BtS classes. [Git branch](#)

AI functions that operate on a city or unit parameter take that parameter as a pointer or reference (`const` when appropriate) to a CvCityAI or CvUnitAI object. Example: `CvUnitAI::AI_allowGroup(CvUnitAI const&, ...)`
 	
Non-AI functions take pointers or references to base-class objects instead (as in BtS). AI functions generally return pointers or references to AI objects. The caller should upcast such a return value into a base-class pointer or reference variable unless the caller is also an AI function. Non-AI functions return pointers and references to base-class objects. Containers of units and cities usually have two sets of accessor functions – those from BtS, returning a pointer or reference to a base-class object, and copies of those functions with "AI_" prepended to their name that return a pointer or reference to an AI object. Example (CvPlayerAI):
`inline CvUnitAI* AI_firstUnit(int *pIterIdx) const {`
`    return m_units->beginIter(pIterIdx);`
`}`
– alongside the BtS function `CvPlayer::firstUnit` that returns a `CvUnit*` from that same `m_units` list.
 	
(All the above mostly also applies to CvSelectionGroup/ CvSelectionGroupAI.)
	While the non-AI classes are abstract and, as such, never instantiated, the interfaces frequently use pointers to those abstract classes; e.g. `CvUnitAI::AI_allowGroup` takes a `CvUnit const*` argument and `CvPlayer::firstUnit` returns a `CvUnit*`. Pointers to CvUnitAI, CvCityAI and CvSelectionGroupAI are very rarely used.

Rationale	So that AI functions can call other AI functions (BtS ensures that through virtual function declarations; see below) and so that non-AI functions can't easily call AI functions. The code duplication around containers is unfortunate, but it really isn't much code all in all and it's not code that is likely to be modified. As an alternative, the containers could always return (pointers/references to) AI objects and leave it up to the caller to store the return value in either an AI or non-AI variable, but upcasting would require the caller to include the header file that defines the relation between base class and derived AI class (i.e. `CvUnitAI.h` or `CvCityAI.h`). I want to avoid including AI headers in non-AI code.
(no change)	All AI functions have an "AI_" prefix attached to their name.
Rationale	I guess this is the Firaxis approach for making programmers aware when they mix AI and game rule code.
Tbd.	If my approach for separating AI/ non-AI code were fully implement (it isn't; see players and teams below), then there wouldn't be much an argument for the name prefixes anymore. Removing them wouldn't necessarily be a big task; mostly automated search and replace.
Added an inline wrappers named simply "AI" for downcasting pointers and references to non-AI objects to references to AI objects. E.g.	Downcasts are (almost?) entirely avoided through virtual function declarations (see below).

`CvCityAI& CvCity::AI()`	
Rationale	While there should be a bit of a barrier against non-AI code calling AI functions, such function calls have to happen sometimes due to the basic (Firaxis) design. An explicit downcast would be too unidiomatic and cumbersome (maybe even unsafe – it's a `reinterpret_cast` if the respective AI header file isn't included). Hence the wrappers. Example: When `CvCity::alterWorkingPlot` needs to update the AI citizen assignment, it does so by calling `AI().AI_addBestCitizen(…)`.
Credits	Nightinggale suggested the wrappers [here](#).

Any AI functions by AdvCiv are non-virtual functions, and I've turned any virtual functions added by K-Mod into non-virtual functions. Same for all the virtual AI functions in CvCity and CvUnit (the EXE, apparently, calls none of those). The virtual AI functions from the original code I've turned into mere wrappers that call non-virtual functions. I've moved the declarations of the wrappers into the private section of the base classes and appended "External" to the function names. For the wrapper implementations, I've created a new file `CvVirtualWrappers.cpp`.	All AI functions are declared as pure virtual functions at the abstract base class. All virtual functions can, in theory, be called by the EXE through the vtable, and the EXE indeed calls some of virtual functions in CvPlayer, CvTeam, CvGame and CvSelectionGroup. (One can't be absolutely certain that the EXE does *not* call some particular virtual function, but if a such a call is never observed in any game mode or menu, then it's a pretty safe bet.)

Rationale	I don't need to call AI functions through a base class pointer or reference. Having to declare each AI function in two places and keeping the declarations consistent is tedious and error-prone. So I'd like to just get rid of the pure virtual declarations, but the vtable position of those functions that the EXE calls mustn't be changed. This means that most of the virtual functions need to be preserved. For a call through the vtable to succeed, it seems that, apart from the table position, only the data types of the parameters must be compatible with the data that the EXE provides. So, changing the accessibility and function names is fine. I've done that to make sure that the wrappers aren't called within the DLL. With the wrappers in place, the non-virtual functions can safely be modified; the compiler will then point out that the call in the wrapper function also needs to be modified. (The wrapper then acts as an adapter.) It's not generally safe to add any virtual functions. There should also be no need; but if need be, then any of the wrappers that the EXE doesn't call can be replaced with a new virtual function.
See also	About deleting some of the virtual wrappers, see comments in `CvVirtualWrappers.cpp`. About adding data members to CvSelectionGroup (kind of related): [003k](#)

`CvGlobals::getGame` returns a `CvGame&`. A new function `CvGlobals::AI_getGame` returns a `CvGameAI&`. For better or worse, players and teams are still usually passed as ids. I've renamed the static functions to `AI_getPlayer` and `AI_getTeam` and created counterparts `getPlayer` and `getTeam` at CvPlayer and CvTeam that return non-AI references. I have not created separate wrapper macros for AI and non-AI references. Instead, `CvGamePlay.h` defines `GET_PLAYER` and `GET_TEAM` as wrappers of `CvPlayer::getPlayer`	The singleton CvGameAI object is accessed (only) through `CvGlobals::getGame`, which returns a `CvGameAI&`. Players and teams are usually passed to functions as enum type ids and the macros `GET_PLAYER` and `GET_TEAM` map those ids to CvPlayerAI and CvTeamAI references. The macros are shorthands for static functions `CvPlayerAI::getPlayer` and `CvTeamAI::getTeam`.

and `CvTeam::getTeam` and `CoreAI.h` redefines the macros (with precedence over `CvGamePlay.h`) as wrappers of `CvPlayerAI::AI_getPlayer` and `CvTeamAI::AI_getTeam`. This means, that access to AI references requires the inclusion of `CoreAI.h`.	
In functions that make more than two function calls on the same player or team object, I usually store a reference in a variable. If I don't call any AI functions, I give that variable a non-AI type.	References to player, team and game objects are rarely stored in variables; instead, `GET_PLAYER`, `GET_TEAM` and `getGame` are called over and over.

Rationale	Returning only AI types forces client code to include the respective AI headers; don't want that when the client function isn't an AI function.
	I was tempted to introduce new macros "PLAYERAI" and "TEAMAI" for AI references, but figured that it's just too much work to change all those call locations. The redefined macro provides only minimal separation between AI code and non-AI code.

See also	See [advc.make](#) about the CvGamePlay, AICore headers.
	[003](#) allows the `GET_TEAM` macro to take a `PlayerTypes` parameter and adds a `TEAMID` macro. And passes CvArea objects always as references or pointers and never by id.

Tbd.	I see no good reason to pass and return players and teams by id. Type information is lost that way (AI/non-AI, `const`) and the macro calls make the code harder to read. It's a good thing that most classes have `getID` functions because ids *are* needed for several purposes (serialization, Python interface, `std::sort`), but it's much more common that member functions need to be called. So all function signatures should be changed from `PlayerTypes` and `TeamTypes` to `CvPlayer const&` and `CvTeam const&` (or a less strict type when necessary). This can be done little by little.
	Afterwards, if PlayerTypes and TeamTypes are still commonplace, one could consider adding pairs of const/non-const functions `CvPlayer::team` returning a CvTeam reference; and `CvPlayerAI::AI_team` returning a CvTeamAI reference – to shorten code like `GET_TEAM(kPlayer.getTeam())`.
	Same situation, by the way, with CvInfo ids in function signatures. That said, the CvInfo classes lack `getID` functions; will have to be added before starting to refactor the signatures.

The life cycle of CvPlayer and CvTeam can't really be changed because the EXE is involved. I'll just document how it works on the right.	All serializable classes are also reusable, i.e. they have reset functions that return the object to a blank state. For CvPlayer and CvTeam, the life cycle of an instance plays out as follows:
	19 objects (`MAX_PLAYERS`) are created as soon as the mod starts loading: The EXE calls `CvGlobals::init`, which calls the `initStatics` functions, which allocate the player and team arrays, call the CvPlayer and CvTeam constructors and assign the array index as the player or team id between 0 and 18.
	The constructors allocate memory for array data members. (Not for all of them; I won't bother to describe that in detail because [advc.enum](#) implements lazy allocation for all member arrays.)
	`reset` and `AI_reset` are called from the base constructors and call `uninit` and `AI_uninit` – which get mostly obsoleted by advc.enum. The reset functions then initialize most data members to some blank value, usually 0 or -1.

	While setting up a new game (e.g. Custom Game screen), CvInitCore may call some player and team functions, e.g. `CvPlayer::updateHuman`. When a new game is started, `init` gets called from the EXE, which calls `reset` once more, properly initializes the data members, in particular sets the alive status (and e.g. `CvPlayer::init` processes leader traits) and calls `AI_init`. The final initialization steps only happen for players and teams alive – once the EXE calls `CvGame::setInitialItems`. When the game is saved, the EXE calls the virtual and overridden `write` function of the derived class, which calls the `write` function of the base class. Likewise, when loading a game, the EXE calls the `read` functions. Before reading from the byte stream provided by the EXE, `reset` and `AI_reset` are called. When returning to the main menu, all 19 objects are `reset`. Only when exiting to the desktop, destructors are called, which free any dynamic memory, either through the `uninit` function or directly. (advc.enum takes over most of the memory managment.)
Rationale	My best guess as to why the player and team objects are created upfront and reused is that the original developers wanted to store information about slot and team assignment in those objects during the setup of a new game. Not a good reason I think, but there's no changing it now.
Tbd.	Perhaps at least the interaction between base class and derived class could be straightened out a bit. Also, the `reset` call from `init` is mostly unnecessary (only the `!bConstructorCall` branch needs to be executed).
See also	003q refactors `CvPlayer::init` and its subroutines.

Cities, units and groups are still managed by FFreeListTrashArray as described on the right. (I'm describing it in some detail because that design pattern is alien to me.) I've merged the `reset` functions of CvCity(AI) and CvUnit(AI) into the `init` functions and made `init` a virtual function overridden by the derived AI classes.

CvCity, CvUnit and CvSelectionGroup have the same functions as CvPlayer and CvTeam for initialization and cleanup, but their life cycle is quite different. In particular, they exist only during a running game. All instances are stored as elements of FFreeListThrashArray (short: FLTA) objects owned by CvPlayer objects. (A comment in that class says "Firaxis Game Engine", so I suppose that's what the initial "F" stands for.) The FLTA class is responsible for allocating, deallocating and serializing its elements.

The creation of a city, unit or group is always initiated from within the DLL. CvPlayer provides factory functions `initUnit`, `initCity` and `addSelectionGroup`, which in turn call `FLTA::add`. `add` calls the respective default constructor, which calls reset functions for a blank state.

`add` also stores a pointer to the element and stores an id *at* the element by calling its `setID` function. `FLTA::getAt` can map that id very effiently to the element pointer. The id is only guaranteed to be unique within the given FLTA. As a globally unique id, the IDInfo struct is used (most importantly in serialized CLLinkLists), which couples the FLTA-internal id with the id of the player that owns the FLTA. Lookup of an IDInfo

happens through e.g. `getCity` (CvGameCoreUtils) via `CvPlayer::getCity`.

Once `add` returns, the factory function (or, in the case of CvSelectionGroup: `CvUnit::joinGroup`) calls `init` on the blank object, providing some crucial data such as plot coordinates.

The destruction of a unit, city or group gets initiated through a `kill` call on that object. `kill` calls a `delete`... helper function on CvPlayer (`deleteSelectionGroup`, `deleteUnit`, `deleteCity`), which `calls removeAt` on the FLTA. The FLTA calls the element's destructors, which call `uninit` and `AI_uninit`. Finally, the FLTA removes the element pointer from its internal array.

When saving or loading, CvPlayer calls the `write` or `read` functions of its FLTAs, and those functions call the `write` and `read` functions of the elements stored in the FLTA. The element functions call `reset`/ `AI_reset` before reading data from the byte stream.

When exiting to the main menu or desktop, `CvPlayer::uninit` calls `FLTA::uninit`, removing all elements.

Rationale	Cities, units and groups don't need to be reusable, so there is no need for `reset` functions. I would've liked to also merge the `init` functions into the constructors, but FLTA requires a default constructor. A virtual `init` function is consistent with the virtual `read` and `write` functions and avoids calling the `init` function of the derived class from the base class (which isn't how inheritance is supposed to work). I haven't made the same changes for CvSelectionGroup because `CvSelectionGroup::reset` is called from the EXE (DLL export) and I don't know under which circumstances that happens.
See also	See advc.agent about the player and team count being hardcoded (and how that could be changed). billw2015 has experimented with replacing the FLTA with a hash map and reported inconclusive results when it came to performance. Git issue

Removed the unused sibling class and merged FLTA with its base class. Moved all code that calls functions of the template argument into an implementation file with explicit instantiations for all template arguments used in the game core code. Added an optional second template parameter so that the first parameter can be instantiated with an abstract base class (CvUnit, CvCity, CvSelectionGroup) and the second with a concrete derived class (CvUnitAI, CvCityAI, CvSelectionGroupAI). The FLTA functions for iteration (`beginIter`, `nextIter`, `getAt`) then return a pointer to an abstract (non-AI) object and counterparts `AI_beginIter`, `AI_nextIter` and `AI_getAt` return a pointer to an AI object.	The FLTA (see above) is implemented as a header-only class derived from an abstract base class. The code base includes an unused class FFreeListArray derived from the same base class as FLTA. All those classes take a type parameter, which has to be a concrete serializable class with `getID` and `setID` functions. For the three FLTA data members of CvPlayer, the type parameters are CvUnitAI, CvCityAI and CvSelectionGroupAI.

See also	Comments in `FFreeListTrashArray.h`

	Caveman2Cosmos also merges FLTA with its base class: [Git commit](#)
Rationale	The problem with the Firaxis design is that all files that iterate over an FLTA need to include the header file that defines the template argument, e.g. `CvCityAI.h`, – because `FLTA::getAt` calls `getID` on the template argument and the implementation of `getAt` gets included as part of the FLTA header. More abstractly speaking, I think the root of the problem is that FLTA is more closely coupled with its elements than container classes normally are. That doesn't necessarily make it a bad design; the problem with header inclusions can be solved by moving the `getAt` implementation (which is too complex to be inlined anyway) out of the FLTA header. (Which requires explicit instantiations, but writing those really doesn't bother me.)
	The remaining dependency problem is the result of FLTA working only with types that it can instantiate and the AI classes being derived from abstract base classes. In BtS, `FLTA<CvUnitAI>::getAt` returns a `CvUnitAI*`. Non-AI code can't implicitly cast that to a `CvUnit*` without including `CvUnitAI.h`. Adding a second type parameter to FLTA is a somewhat clumsy solution (that also involves explicit casts in the FLTA header), but at least it encapsulates the problem and allows for clean client code.
	I've merged FLTA with its base class mostly to make the changes above easier to implement. The virtual functions weren't suitable for inlining anyway, so performance wasn't a rationale. That said, removing the FLTA header dependencies has allowed me to inline all the iteration wrapper functions at CvPlayer, CvGame (`FLTA<CvDeal>`) and CvMap (`FLTA<CvArea>`), e.g. `CvPlayer::nextUnit`.
Tbd.	I feel that it should be possible to implement the `getAt` function without any conditionals; or perhaps a more narrow function that can replace most of the `getAt` calls. Could the `nextIter` function be inlined?
	If CvUnitAI is going to have derived classes (see *Tbd.* higher up), then a class CvUnitList should be derived from `FLTA<CvUnit,CvUnitAI>` with a factory function `add(DomainTypes)` that calls a `protected` function `FLTA::add(AITypes*)` to insert the newly created element. And `CvUnitList::add(void)` should `FAssert(false)`. That way, FLTA can maintain full ownership of its elements despite both template parameters being abstract classes.

In `CvUnit.cpp`, I've replaced any `getUnitInfo` calls with direct accesses to `CvUnit::m_pUnitInfo`. But in `CvUnitAI.cpp`, I've replaced all direct `m_pUnitInfo` accesses with `getUnitInfo` calls.	`m_pUnitInfo` gets accessed quite frequently and it's usually done without a `getUnitInfo` call.
Rationale	This way, it'll be easier to move CvUnitAI code into component classes that aren't derived from CvUnit. (No concrete plans for that.)

003v	Don't load unused XML data

AdvCiv	*BtS*
The XML files for random events aren't loaded until a game is started or loaded that has random events enabled. Similarly, the XML files for the throne room aren't loaded until the player accesses the Throne Room screen.	All XML files are loaded either when BtS (or a mod) is launched or when a game is started (new game or saved game). Random events and throne room are loaded at game start regardless of game options. (The Throne Room screen is an unfinished screen from Vanilla Civ 4 that can be accessed via the Debug menu – Ctrl+Shift+D in Debug

	mode.)
Rationale	To speed up XML loading (many players disable random events) and save memory. However, the speedup is just a (few?) hundred milliseconds faster and the memory savings are probably also entirely negligible.
See also	003i: The most effective way to speed up XML loading would be to get the XML cache to work.

003w	Layer in between XML data classes and game objects

AdvCiv	*BtS*
Turned all functions in CvGameCoreUtils that operate solely on CvInfo objects into member functions (some static) of CvInfo classes.	The CvInfo classes are pure data classes, which is to say that their interfaces correspond almost exactly to the structure and tag names of the XML files. For the most part, the classes that represent the game state and the AI work directly with that interface. In some cases, global functions in CvGameCoreUtils act as an intermediate layer, for example `isTechRequiredForBuilding(TechTypes, BuildingTypes)` or `isWorldWonderClass(BuildingClassTypes)`.

See also	Comments in `CvGameCoreUtils.h` say how the removed functions have been replaced.
Rationale	It should be commonplace that the DLL adapts the XML-derived interface, so it's not something that should be handled by a handful of global "helper" functions. My tentative approach is to include that extra layer (or "enhanced interface") in the CvInfo classes until enough functions have accumulated to justify a separate class.
	Also, functions in CvGameCoreUtils can rarely be inlined without messing up header inclusions. (`CvGameCoreUtils.h` is part of the precompiled header.)

New class CvCivilization that handles the mapping between unit and building classes and unit and building types. CvCivilization precomputes the units and buildings that a civilization can ever produce.
 	
Replaced many loops over all units or buildings that a particular player owns or considers to produce with e.g.:
 	
``CvCivilization const& kCiv = getCivilization();
for (int i = 0; i < kCiv.getNumUnits(); i++)
{
   UnitTypes eUnit = kCiv.unitAt(i);```
	The terminology, just to be clear, is that e.g. Axeman is both a unit class and the default unit type for that class, while Vulture is a unique unit type of the Axeman class.
	
	Loops over all units (similar for buildings) usually take either the form (`CvCity::canTrain(UnitCombatTypes)`)
	
	``for (int i = 0; i < GC.getNumUnitClassInfos();
	   i++)
	{
	   UnitTypes eUnit = (UnitTypes)
	       GC.getCivilizationInfo(
	       getCivilizationType()).
	       getCivilizationUnits(i);
	   if (NO_UNIT != eUnit)
	   { // ...```
	
	or simply (`CvPlayerAI::AI_bonusTrade`):
	
	```for (iI = 0; iI < GC.getNumUnitInfos(); iI++)```

See also	Comments in CvCivilization.h
Rationale	Mainly to make unit and building loops easier to read.

	I had also hoped that replacing some of the loops over all unit or building types would yield a bit of a speedup, but there's no real difference. (If a mod-mod were to add more unique types, that could change.)
Credits	Nightinggale's [CivEffects](#) class in "We the People" has been an inspiration. That class also forms a (cache) layer between XML data and game state – though CivEffects deals with player-specific effects from various sources. If that concept were applied to BtS, then e.g. leader traits would be "civ effects" and wonder abilities like Notre Dame's +2 happiness in the owner's cities would also be "civ effects".

003x	Changes to the design of the CvInfo classes	
AdvCiv		*BtS*
Split up `CvInfos.h` into 18 headers. Some of those include each other though and many are included in `CvGameCoreDLL.h`; so the the overall number of include directives in the code hasn't increased much. Split `CvInfos.cpp` into 17 implementation files.		`CvInfos.h` essentially defines one class for every non-schema XML file in the subfolders of `Assets\XML`. The BtS version of `CvInfos.cpp` has more than 20 000 lines, making it the largest implementation file (though not by far; in AdvCiv, `CvPlayerAI.cpp` has always been larger).
Rationale	To make the CvInfo code easier to work with, to reduce compilation time and to encourage encapsulation. For rationales for this particular way of partitioning `CvInfos.h`, see the comments in the new `CvInfo_*.h` files.	
See also	Caveman2Cosmos has also split up CvInfos (in a coarser way), and has been considering a more fine-grained split ([Git commit](#)) that I took some inspiration from.	
Tbd.	Perhaps move CvUnitInfo from `CvInfo_Unit.h` (which also includes e.g. CvPromotionInfo) into a separate header, and perhaps include that header in `CvGameCoreDLL.h`. Currently, `CvInfo_Unit.h` gets included in `CvUnit.h` for inlining and CvUnit.h itself is frequently included in implementation files, so the whole `CvInfo_Unit.h` gets recompiled frequently, but only CvUnitInfo is actually needed by `CvUnit.h`.	
Removed unhelpful comments (i.e. most comments) from CvInfo code. Removed empty default constructors and destructors. Replaced pairs of `FAssertMsg` calls with a single `FAssertBounds` call. Replaced `if`/`else` sequences with `switch` in the implementation of CvActionInfo.		
Rationale	To unclutter the CvInfo code (while I was at it moving the code around).	
Removed the unused `iInfoBaseSize` parameter from the `CvXMLLoadUtility::SetVariableListTagPair` functions.		
Rationale	To remove unnecessary dependencies on the info classes.	
Replaced integers in some of the CvInfo member function signatures with enum types. Preserved the old signatures in private wrapper functions that are exposed to Python.		The CvInfo signatures don't use enum types at all. Presumably, to avoid having to write a Python wrapper class for each CvInfo class to handle the casts. (This doesn't really explain the absence of enum return types, which aren't a problem for Python.) This leads to frequent explicit type casts in DLL code that calls the CvInfo functions.
Tbd.	Only CvBuildInfo fully and CvBuildingInfo and CvUnitInfo partly done so far.	
Rationale	Return types: Encapsulating the int-enum casts in the CvInfo classes leads to cleaner	

code everywhere else.

Argument types: Improved type safety. Due to the changes under advc.enum, the caller knows the correct enum type most of the time (or the surrounding code should be easy to refactor accordingly).

See also	See `CyInfoWrapper.h` about the Python wrappers.
	"We the People" Git issue that proposes to use enum types in signatures where possible (not specifically in the CvInfo classes – anywhere). Should take care of that through macros that define enum maps and their (typesafe) accessor functions, see advc.003t.

003y	Separate class for DLL-to-Python calls	
AdvCiv		*BtS*

New class CvPythonCaller with essentially a separate function for every type of Python call. Example:

```
bool CvCity::canCreate(ProjectTypes eProject,
   bool bContinue, bool bTestVisible) const
{
   if (GC.getPythonCaller()->
      canCreateOverride(*this, eProject,
      bContinue, bTestVisible))
   return true;

   if (!GET_PLAYER(getOwner()).
      canCreate(eProject, bContinue,
      bTestVisible))
   return false;

   if (GC.getPythonCaller()->
       cannotCreateOverride(*this, eProject,
       bContinue, bTestVisible))
      return false;

   return true;
}
```

Python calls in the DLL use a fairly low-level API, which results in verbose code, for example:

```
bool CvCity::canCreate(ProjectTypes eProject,
   bool bContinue, bool bTestVisible) const
{
   CyCity* pyCity = new CyCity((CvCity*)this);
   CyArgsList argsList;
   argsList.add(gDLL->getPythonIFace()
      ->makePythonObject(pyCity));
   argsList.add(eProject);
   argsList.add(bContinue);
   argsList.add(bTestVisible);
   long lResult=0;
   gDLL->getPythonIFace()->
       callFunction(PYGameModule, "canCreate",
       argsList.makeFunctionArgs(), &lResult);
   delete pyCity;
   if (lResult == 1)
      return true;

   if (!(GET_PLAYER(getOwnerINLINE()).
      canCreate(eProject, bContinue,
      bTestVisible)))
   {
      return false;
   }

   pyCity = new CyCity((CvCity*)this);
   CyArgsList argsList2;
   argsList2.add(gDLL->getPythonIFace()
      ->makePythonObject(pyCity));
   argsList2.add(eProject);
   argsList2.add(bContinue);
   argsList2.add(bTestVisible);
   lResult=0;
   gDLL->getPythonIFace()
      ->callFunction(PYGameModule,
      "cannotCreate",
      argsList2.makeFunctionArgs(), &lResult);
   delete pyCity;
   if (lResult == 1)
      return false;

   return true;
}
```

Rationale	Mainly to make the Python calls – many of them callbacks that are unused on the Python side – less visible. They really clutter the DLL code in BtS.

See also	[003](#) similarly moves some logging code into a separate class (CvDLLLogger). Caveman2Cosmos instead takes the approach of making the generic DLL-to-Python interface more compact. [Git commit](#) (perhaps not the only one) C2C also has started to get rid of the `isNone` functions by letting C++ functions return `NULL` instead of dummy objects. This removes a lot of clutter on the C++ side, but also requires some changes to Python code (replacing all `isNone` calls with built in `is None`). [Git commit](#) (again, there may be more) And it seems that C2C handles the int-enum casts between DLL and Python more intelligently: [Git pull request](#)

By default, CvPythonCaller asserts that the Python function was successfully called. Found out about two (unused) missing Python functions this way; removed the one (`doHeadquarters`) on the DLL side, added the other (`cannotSpreadReligion`) on the Python side. Copied `CvGameInterface.py` into the mod in order to add `cannotSpreadReligion`.	A Python call fails if the target function doesn't exist in Python or if the call parameters don't match the Python signature. This can be normal in the case of map scripts. Normally, it's a bug and the BtS DLL code checks for such bugs only sometimes. `CvGameInterface.py` is mostly (but not entirely) obsoleted by BUG.
Instead of CvGlobals, CvPythonCaller handles the caching of the callback guards. This is now done through an array of enum values. The guards for Python events are now handled by the CvDllPythonEvents class. Added some more callback guards – though the gains in performance are negligible (1% or so).	The callback guards are defined in `PythonCallbackDefines.xml` and all set to 0, meaning that the DLL should always skip the respective Python call in order to save time. Example: `USE_CAN_TRAIN_CALLBACK`=0 disables the Python call to `CvGameInterface.canTrain` in `CvCity::canTrain`. Python modders that want a guarded Python call to happen (despite a slight performance penalty), can disable the guard through the XML file. On the DLL side, each guard is stored as an individual boolean data member of CvGlobals. K-Mod has added several more callback guards.

See also	[003b](#) uses the same technique for GlobalDefines cached by CvGlobals.
Rationale	Part of the effort to move the Python stuff out of sight. I don't think I'll ever want one of those guarded functions to be called – they're for modders who want to change the game rules and aren't able to recompile the DLL.

AdvCiv	*K-Mod*
Moved the computation of game score and capture gold from `CvGameUtils.py` to the DLL (`CvPlayer::doCaptureGold`). Moved the code that triggers the "Partisans" event from `CvEventManager.py` to the DLL (`CvCity::doPartisans`) and slightly changed it so that units spawn for the civ with the highest tile culture instead of the one with the highest city culture. The Python code also contained a bug that had caused partisans to appear for size-1 cities; fixed.	K-Mod had already moved the pillage gold and level-up threshold computation from Python to `CvUnit::pillage` and `CvUnit::experienceNeeded`.

Rationale	Not sure why these had been implemented in Python; perhaps as instructional examples for Python modders. I might want to change those formulas though, and

	that'll be easier (for me) in C++, and faster.
See also	The change to the partisan event was prompted by [this]() old CFC post by DanF5711. I accidentally fixed the error in the Partisans code and learnt about it from [this]() CFC post by SmokeyTheBear. That same user found another bug in `getNumPartisanUnits` (`CvRandomEventInterface.py`); tagged with change id [001]().

advc.enum	Enum map classes, traits for global enum types and related code
Rationale	Memory optimization (for improved CPU cache performance); improved code readability, extensibility. Well, the enum map and traits code is highly reliant on templates and as such far from being easy to read; but the code that employs enum maps replaces a great volume of error-prone boilerplate dealing with naked arrays. In a perfect (object-oriented) world, much of the two-dimensional data would be handled by component classes. For example, a class that represents all knowledge that one (AI) team has about another team instead of a bunch of data structures that map a team id to one particular statistic about that team.
Credits	The enum map classes are based on Nightinggale's [EnumMap]() class in the "We the People" mod (WtP). wrote all the code. (I made a minuscule early contribution by [suggesting]() the class name "EnumMap".)
See also	[003t]() integrates enum maps with XML loading code.

AdvCiv	BtS
Use enum maps for storing mappings from one or two enum types to some other – usually integral – type. This concerns multi-dimensional data loaded from XML, class members that store the (serializable) game state and, at times, local variables and function arguments. One type of enum map, `ListEnumMap`, keeps a list (implemented as a resizable array) of only those key-value pairs whose value differs from a default value. The default value is set at compile time through a template parameter (as in WtP). This data structure allows for fast iteration over the non-default data, and, to this end, there are macros `FOR_EACH_NON_DEFAULT_PAIR` and `FOR_EACH_NON_DEFAULT_KEY`. Finding the value for a (single) given key involves traversing the list of pairs, so that operation is not super fast.	Uses two- and three-dimensional arrays. Memory for those arrays is, in most cases, allocated in constructors or functions called from there. (Notable exception: CvPlayer and CvTeam allocate their arrays upon being set to "alive" status.) Most of the arrays have serialization code in one of the `read`/`write` functions, and are accessed in `get`, `set` and `change` functions with assertions that check the array bounds.

The other essential type of enum map is `ArrayEnumMap`, which stores a value for every possible key in a non-resizable array. Lookup (random access) is fast, but large arrays can negatively affect the performance of the CPU cache. Two optimizations to reduce memory use (both from WtP) allocate the array lazily upon setting a non-default value, and store boolean values in a bit array. AdvCiv allows these optimizations to be disabled through template parameters. (Lazy allocation is counterproductive when non-default values are guaranteed to be stored.) Another WtP optimization supported by `ArrayEnumMap` is the use of automatically allocated memory when the array requires only a few byte. This avoids overhead for dynamic allocation (especially helpful for local variables) and improves memory locality.

For the list-based approach, I've implemented bit arrays only for the special case of mappings to `bool` loaded from XML. Mappings to `bool` can be interpreted as (mathematical) sets, so I've

named the respective enum map type `OfflineListEnumSet`.

Another WtP optimization, the use of two-byte or single-byte types for storing values of an enum type, is implemented for all enum map types. Moreover, any (sensible) data type can be specified for the internal storage of the value type. For example, a map storing the yield rates of a tile can use `int` in its public interface and use `char` internally. Assertions to ensure that the limits of the internal type aren't exceeded can be enabled through a preprocessor switch.

For three-dimensional data, i.e. mappings from pairs of enum keys to (single) values, there's an `EnumMap2D` class that maps the first (outer) key type to (a pointer to) an inner enum map, which in turn maps the second (inner) key type to the (inner) value type. This is similar to the `EnumMap2D` class in WtP – but more flexible insofar as `ListEnumMap` and `ArrayEnumMap` can be freely combined for the outer and inner map type.

Often, one of the key types is a yield or commerce type and the value represents a yield or commerce rate, rate change or percent modifier. In that case, the mapping can be interpreted as associating an outer key type with a 3- or 4-tuple of yield or commerce statistics small enough to be encoded in a single (unsigned) `int` or `long long int` value. The `Enum2IntEncMap` class uses a simple (non-nested) enum map internally (avoiding dynamic allocation of inner enum map instances) and provides the same interface as `EnumMap2D` externally.

The subscript operator has been overridden – but only for `ArrayEnumMap`, so it's generally better to stick to the get, set and arithmetic functions so that an `ArrayEnumMap` can be easily replaced with a `ListEnumMap`.

See also (cont.)	Comments in `EnumMap.h`. C2C has a class [IDValueMap](#) that fulfills a similar role as my ListEnumMap. IDValueMap uses `std::pair` internally. I use two separate arrays for keys and values (but had used a single array of `std::pair` in an earlier version of my code; I don't think there's a significant difference in performance). C2C also stores some boolean arrays as simple lists (vectors); [SVN revision](#). Replacing CvArea member arrays with EnumMaps has resulted in a [speed-up of almost 5%](#), in part, because change [030](#) tends to increase the number of separate water areas on the map. For CvPlot, WtP uses an additional data structure called "[RevealedPlotData](#)". The premise is that the route and improvement revealed to a team are "usually used together." I think that's not really true for the AdvCiv code. Routes are important for movement, improvements for tile yields; there isn't much code that deals with both. karadoc had started (slowly) to replace some arrays with vectors: [Git commit](#) [advc.fract](#): The ScaledNum class is fully supported as a value type for enum maps. ([This CFC post](#) by Nightinggale advising me on how to integrate ScaledNum with his EnumMap class is outdated.)
Tbd.	CvGame, CvPlayerAI, CvCityAI and essentially all CvInfo classes except those defined in `CvInfo_Building.h` still use arrays instead of enum maps. Just haven't gotten around to refactoring them yet. Searching the code base for all occurrences of "new int" or "new bool" should identify all classes that still need work. I had implemented a `ListAndArrayEnumMap` class storing data redundantly in order to maximize the speed of both random access and iteration, but I haven't bothered to reimplement it after merging my CvInfoEnumMap hierarchy (aimed only at data from XML) with the "We the People" EnumMap class. Could still bring this class back if a use arises – but I doubt that it will. The latest source code is here: [GitHub](#)
Rationale (cont. –	Apart from sparse data loaded from XML, list-based enum maps are important for per-civ data stored at CvPlot. Some of that data, e.g. data about nearby cities, gets

list-based maps)	allocated for a large portion of plots, but is used only for a small number of players. When the civ count is increased beyond 18, the unused memory allocated by the (array-based) WtP enum map (or the BtS arrays) seemed to have a significant impact on cache performance. In a test with the Earth18 scenario, a DLL allowing 31 civs (i.e. 32 players including the Barbarians) had about 28% longer turn times than one allowing 18 civs if only WtP enum maps were used for the CvPlot member data. After converting some of the enum maps to `ListEnumMap` (to a precursor named "SparseEnumMap" – to be exact), this performance penalty for allowing (but not using) thirteen additional civs decreased to 9.5%. Earlier tests suggest that the performance penalty is smaller on Huge random maps (which have smaller dimensions than Earth18); it was 8% the last time I tested it and might be as low as 5% now, which would mean that AdvCiv can switch to a 31-civ DLL whenever (or if ever) I'm comfortable with breaking savegame compatibility.
See also (cont.)	056 facilitates tests like the one with Earth18: scenarios no longer need to contain data about every civ id that the DLL recognizes. advc.agent has also helped reduce the performance penalty for unused civs.
Tbd. (cont.)	Experiment with some other per-civ data. That said, no class looks particularly promising for this. CvArea has a lot of per-civ data, but most of it doesn't get allocated for uninhabited areas. Perhaps some more CvPlot data – through the preprocessor, we can use an array-based enum map when the civ limit is 18 and a list-based one when it's (considerably) higher.

The enum map classes are based on a type traits system that associates most of the global enum types (`CvEnums.h`) with information about its (maximal) length and with the smallest (signed) integer type that can safely store all values of the enum type.

The enum traits are mostly generated by the preprocessor based on type lists in a new header `CvEnumMacros.h`. That header also provides increment operators for most enum types and several macros described below.

A header for arithmetic type traits treats (global) enum types and ScaledNum as arithmetic and thus allows the enum map classes to apply simple arithmetic operations on enum and ScaledNum values.

AdvCiv	*WtP*
The enum map classes are based on a type traits system that associates most of the global enum types (`CvEnums.h`) with information about its (maximal) length and with the smallest (signed) integer type that can safely store all values of the enum type. For enum types whose length is only known at runtime, the integer type is hardcoded, but assertions after XML loading verify that those integer types are indeed sufficiently large. The enum traits are mostly generated by the preprocessor based on type lists in a new header `CvEnumMacros.h`. That header also provides increment operators for most enum types and several macros described below.	To determine the internal array size (and the number of bytes to store per value when mapping to an enum type), the WtP `EnumMap` obtains the number of enum values from a `NUM_..._TYPES` enumerator at the end of every supported enum type. Those enumerators are generated by an external Perl script that parses the "info" XML files. As a result, adding any `type` elements to an XML file requires the game core DLL to be recompiled. For enum types with a dynamic range, the length needs to be hardcoded by defining an `ArrayLength` function and a byte size (1 or 2 bytes).
Rationale	I don't want to adopt the Perl code generator. It adds another dependency to the build environment (could port the script to Python to avoid that I guess) and requires a special DLL to be deployed for XML modders who are unable to recompile the DLL. My approach is more portable insofar – although not as powerful: the preprocessor isn't as flexible as an external script and length information known at compile time

	allows for better code optimization.
My enum map classes serialize data in (mostly) the same compact (binary) format that they use internally. They can read data in various (less compact) formats that BtS uses for serializing arrays.	Savegame writer class that uses a compact annotated format – for persistent enum maps and all other persistent data.

See also	Nightinggale on the WtP savegame format: [CFC link](#) WtP wiki: [Savegame format](#)
Rationale	Getting the deserialization functions for the BtS format right was a lot of work. If I ever change my serialization functions, it'll again be a lot of work to maintain compatibility with my current format. An annotated format really is the saner approach, but adopting the WtP format is too big a task for now – seeing that it concerns not just enum maps but all persistent data.
Tbd.	If AdvCiv moves beyond version 1.0x, then I'll want to break savegame compatibility eventually, and, at that point, adopting the WtP format should be more feasible.

Macro `FOR_EACH_ENUM` added for iterating over an enum type. Takes an XML type name as its only parameter and defines a loop counter variable eLoop{`Name`}. For cases where "eLoop{`Name`}" is too long because the loop variable gets referenced numerous times, I've added a variant `FOR_EACH_ENUM2` that takes a variable name as a second parameter. Added `FOR_EACH_ENUM_REV` for cases when reversing the order will speed up the search for a particular element.	Macro `FOREACH` that gets used mostly in conjunction with `EnumMap`. (I don't think it's supposed to replace all loops over enum types in the WtP DLL code base.) Those loops have the following form in BtS: ```for (int i = 0; i < GC.getNumBuildingInfos(); i++) { BuildingTypes eBuilding = (BuildingTypes)i;```

Credits	It's mostly Nightinggale's work; see [this](#) WtP Git issue.
Tbd.	Parts of CvGame, CvGameTextMgr, CvPlayerAI, CvUnitAI don't use the macro yet.
Rationale/ Tbd.	As for the lengthy variable name: A nondescript name like "eBuilding" has a good chance of clashing with a variable name in an enclosing scope (no `Zc:forScope` option in MSVC03). Enclosing the loop in curly braces would be possible through a pair of macros (as Nightinggale had contemplated), and that would also allow the macro to define a reference to an info object; e.g. ```FOR_EACH_INFO_START(Building, LoopBuilding) if (eLoopBuilding == ... kLoopBuilding.get... END_FOR_EACH``` Or, optionally with braces. VS IntelliSense can handle such macros, but they're still cumbersome. Note that it's impossible to define an info object reference before the body of a loop because a reference can't be reassigned. Even exposing the vectors stored at CvGlobals wouldn't change that. Another possibility would've been (not going to change that now) to use "el" as the default loop counter name instead of "eLoop..."
See also	[003s](#) defines macros for iterating over lists. All those macros are listed in `cpp.hint` to

	fix problems with IntelliSense.
	: The AgentIterator classes should usually be preferred over `FOR_EACH_ENUM` when looping over players and teams.

Defined macros `LOOP_INFO(TypeName)`, `SET_LOOP_INFO(TypeName)` for obtaining or defining a reference to an XML info object in the body of a `FOR_EACH_ENUM` loop. These macros call preprocessor-generated functions `CvGlobal::getLoopInfo` that don't check array bounds (not even in assert builds). Examples:
• `SET_LOOP_INFO(Building);` instead of:
`CvBuildingInfo const& kLoopBuilding = GC.getInfo(eLoopBuilding);`
• `LOOP_INFO(Building).getBuildingClassType();` instead of:
`GC.getInfo(eLoopBuilding).getBuildingClassType();`

Commented out for now.

Rationale	The array-bounds assertions are unnecessary when looping over all instances with `FOR_EACH_ENUM`, and they get in the way of /Ob1-inlining in assert builds (which isn't a big concern, but still). Perhaps more importantly, the `SET_LOOP_INFO` macro saves the user from having to type the enum name three times.
	Most `FOR_EACH_ENUM` loops in the codebase don't define a CvInfo reference, but many of those loops obtain such a reference from CvGlobals in one or two places. So it wouldn't be much trouble to deploy the `SET_LOOP_INFO` macro everywhere it makes sense, but doing the same for `LOOP_INFO` seems like too much work. I feel it would be a bit inconsistent to remove the array-bounds checks only from loops where a CvInfo reference is explicitly stored in a variable. Therefore, I'm not using either macro.

`FOR_EACH_ENUM_RAND` macro that goes through the enum types in a random order.

Rationale	Rarely done in the BtS code, but should perhaps be done more often. I'm pretty sure that there are some loops in the AI code where the fixed order causes a bias toward low enum values. With this macro, shuffling takes practically no extra implementation effort. (Though shuffling isn't free in terms of computing time of course.)
See also	: For players and teams, AgentIterator has a randomization parameter.

`FAssertEnumBounds` macro that takes a single enum value as parameter and asserts that it is nonnegative and less than the enum length. `FAssertInfoEnum` is the same except that it also allows `-1`. Both based on enum type traits.

Rationale	One step beyond the `FAssertBounds` macro (see). I don't use it very widely because the CvInfo classes mostly use integer indices (they shouldn't; see *Tbd*. under) and because many bounds assertions have been made unnecessary by enum maps. And for player and team arrays, it's not clear enough whether the upper bound should include the Barbarians.
See also	applies `FAssertInfoEnum` to simple (non-nested) enum elements loaded from XML. (But not when the DLL stores those elements as `int`; again, see .)

AdvCiv	*BtS/WtP*
New (header-only) class CityPlotIterator. Usage example:	WtP uses the `FOR_EACH` macro for those loops; BtS:
``` for (CityPlotIter it(kCity); it.hasNext(); ++it) { CvPlot const& kPlot = *it; CityPlotTypes const ePlot = it.currID(); // ... } ```	``` for (int iI = 0; iI < NUM_CITY_PLOTS; iI++) { CvPlot* pPlot = plotCity(kCity.getX(), kCity.getY(), iPlot); if (pPlot != NULL) { // ... } ```

Can also take a CvPlot parameter and `bIncludeCenter=false` to exclude the center tile. And there are derived classes for excluding unworked or unworkable tiles and randomizing the order of traversal. The enum type "CityPlotTypes" is also new. Replaced all the BtS-style loops with CityPlotIterators and a few with `FOR_EACH_ENUM` (when `NULL` plots aren't supposed to be skipped).	`}` To exclude the center, `iI==CITY_HOME_PLOT` is checked. The efficient way to do that is in the initialization of `iI` (as `CITY_HOME_PLOT` is 0); BtS usually checks it in the body of the loop instead.
See also	The syntax is consistent with [advc.agent](); see rationales there. Fairly detailed comments in `CityPlotIter.h`. [advc.plotr](): An iterator for square areas of tiles.
Rationale	Can't beat the speed of the BtS code, or even match it. The iteration overhead is about two times worse than in BtS, and I think that's the best one can do with an iterator. Speed matters because these loops occur very frequently, but avoiding the `NULL` check (in the client code) and not having to remember three unrelated identifiers (`NUM_CITY_PLOTS`, `plotCity`, `CITY_HOME_PLOT`) is worth the very slight performance penalty.

Enum type `PlotNumTypes` added for the CvPlot ids computed by `CvMap::plotNum`. Stored as `__int16` in enum maps unless `MAX_CIV_PLAYERS` has been increased beyond 31; in that case, `__int32` is used. The upper limit of `__int16` is 32768, so that's now the map size limit (e.g. 256x128 or 181x181) for the 18-civ DLL.	`CvMap::plotNum` returns an `int` and `plotByIndex` takes an `int` argument. I don't think WtP allows plot ids in `EnumMap`. Players have been able to start games with 200x200 tiles and probably more, but no one finishes those games. Apart from unplayable experiments and the C2C Ultimate Earth Map, the biggest scenario that I'm aware of is the Gigantic Accurate Earth Map with 232x112 plots. Huge maps have ca. 50% more tiles than Large maps. If that progression is continued, then two sizes beyond Huge would still have fewer than 25000 plots. SmartMap has a 512x512 setting, but it's [reportedly]() not really playable.
Tbd.	`CvMap::plotByIndex` still takes an `int` argument. If I change that, then all loops that traverse the entire map will have to be revised. For loops that only deal with plot ids (rare), `FOR_EACH_ENUM(PlotNum)` will be adequate; most loops need to go through the CvPlot instances; an iterator should be written for that. "MapIter" could be misunderstood; "MapGridIter"? Some filters (land/ water)? Should be tagged with [advc.plotr]().
Rationale	For now, the enum type is mainly for enum maps, specifically for their use in the starting position calculation ([027]()).

AdvCiv	*BtS*
Functions `CvGlobals::get{TypeName}Info(TypeName)` replaced with overloaded functions `CvGlobals::getInfo(TypeName)`. Except for XML info types that have no associated enum type. The old `get...Info` functions still exist, but they're unused and deprecated.	There is a uniquely named `CvGlobals::get...Info` function for every CvInfo type.

Rationale	To shorten the info accessor calls. It was easy enough to generate the `getInfo` functions from the enum list macros and to change the four-digit number of call locations through automated search and replace – using, for future reference, this regular expression in Notepad++: `GC\.get(....|....|.....|......|.......|.......|.......|..........|.........|...........|............|.............|...............|...............|..............)Info\(`
Tbd.	Should perhaps add enum types for the few info classes that are missing one so that those can also be accessed through a `getInfo` function.

The overloaded `getInfo` functions and other parts of the CvGlobals class are generated by the preprocessor based on the type lists in `CvEnumMacros.h`.	There were several hundred of lines of repetitive code.

Rationale	I was a bit worried about debugging (can't set a breakpoint in a function generated by the preprocessor), but I think the function definitions I removed were so simplistic that there should be no need to step into them.

Global comparison operator functions are declared for certain dangerous combinations of enum types, in particular `PlayerTypes` and `TeamTypes`. No definition is provided for these operators, so that it's impossible to call them (will result in a linker error).	Comparisons between enum types aren't type-safe in C++, and there is nothing in the BtS code to detect errors like `if(kPlayer.getID() == eTeam)`. Modders seem to have fixed all errors of this kind over the years, i.e. my overloaded operators didn't reveal any errors in BtS code (but several in AdvCiv code).

Rationale	A compiler error would be more helpful, but only the linker can tell whether a global function has any call locations.
Tbd.	Should perhaps declare comparison operators for all enum types used in the DLL. I'm not sure if those thousands of declarations would slow down the compiler though. It would be nice to get rid of most of the enum type comparisons, e.g. by writing `if(kPlayer.isTeam(eTeam))` in the example above. Inlining such an `isTeam` function would require `CvTeam.h` and `CvPlayer.h` to include each other. `CvPlayer::operator==(PlayerTypes)` etc. is also worth considering.
See also	It's easy to get the `NO_...` enumerators and `NULL` mixed up, and such an error can go unnoticed for quite some time. I've written code for prohibiting all equality tests between integers and enums (commented out in `CvEnums.h`), but, currently, this would require too many explicit casts. Maybe once the AgentIterator classes are used more widely (see [advc.agent](#)) and once the CvInfo classes use enum types instead of integers (see [003x](#)) when possible. The "We the People" mod makes all int and enum comparisons type-safe through static assertions: [Git commit](#) I don't think the static assertions can work with my approach of forbidding only (certain) enum-enum comparisons; one of the parameters in the comparison would have to be a template type.

Turned the bitmasks that were defined through the preprocessor in `AI_Defines.h` into enums with overloaded bitwise operators and renamed the header to "AIStrategies.h". I've named the enum for the victory strategies "AIVictoryStage" and renamed the `AI_...VictoryStrategy` functions (CvPlayerAI, CvTeamAI) accordingly. Also turned some bitmask definitions in `CvDefines.h` into enums.

Rationale	The two types of strategy defines were easy to confuse; type-safety was badly needed. The name change is also intended to make the strategy types more distinct.

	The old function names were also very long. I understand that C++11 has strongly typed enums, but in MSVC03, overloading the bitwise operators seems to be the best one can do. (I think it's fine.)
See also	I've found three bugs through this change, listed somewhere under [001]().
Tbd.	There are some other uses of bitmasks in the code base that could benefit from the same treatment. Also, bitmasks could be used in additional places, in particular to shorten parameter list, for example of the BestDefender function. C2C does that too ([Git commit]()).

advc.agent	Iterator classes for looping over "agents" – i.e. players and teams
Tbd.	Currently, the caches at `CvAgents` are only updated under very specific circumstances, e.g. when a new colony is created. Will probably have to use more generic update functions (e.g. whenever the "alive" status of an agent changes) to make sure that the caches are up to date during game initialization. So far, I've refrained from using agent iterators in any initialization code, erring on the side of caution. See also comment in `AgentIterator.h`.

AdvCiv	*BtS*
| AgentIterator class with parameters for frequently needed sequences of agents. For example:<br><br>```
for (TeamIter<MAJOR_CIV> it; it.hasNext(); ++it)
{
    CvTeam& kLoopTeam = *it;
    // …
}
```<br><br>(where "MAJOR_CIV" means non-Barbarian, non-minor civ alive) | A loop over all non-minor, non-Barbarian teams alive looks like this (`CvGame::testVictory`):<br><br>```
for (int iI = 0; iI < MAX_CIV_TEAMS; iI++)
{
    CvTeam& kLoopTeam = GET_TEAM((TeamTypes)iI);
    if (kLoopTeam.isAlive())
    {
        if (!(kLoopTeam.isMinorCiv()))
        {
            // …
        }
    }
}
```<br><br>Or written a bit more nicely:<br><br>```
for (int i = 0; i < MAX_CIV_TEAMS; i++)
{
    CvTeam& kLoopTeam = GET_TEAM((TeamTypes)i);
    if (!kLoopTeam.isAlive() && !kLoopTeam.isMinorCiv())
        continue;
    // …
}
``` |
| Example 2:<br><br>```
for (PlayerAIIter<ALIVE,MEMBER_OF> it(getID());
    it.hasNext(); ++it)
{
    CvPlayerAI const& kPlayer = *it;
    // …
}
```<br><br>That can be shortened further:<br><br>```
for (MemberAIIter it(getID()); it.hasNext(); ++it)
{
    CvPlayerAI const& kPlayer= *it;
    // …
}
``` | Example 2: A loop over all living members of "this" team (from `CvTeamAI::AI_calculateAreaAIType`). This actually contained a bug (fixed by the unofficial patch) – the loop index was named "iPlayer", but "iI" from an earlier loop was passed to `GET_PLAYER`; corrected code:<br><br>```
for (int i = 0; i < MAX_CIV_PLAYERS; i++)
{
    CvPlayerAI const& kPlayer = GET_PLAYER(
                        (PlayerTypes)i);
    if (!kPlayer.isAlive() ||
        kPlayer.getTeam() != getID())
    {
        continue;
    }
    // …
}
``` |
| *Rationale* | The iterator syntax is shorter and less error-prone. Range-based loops would be even shorter, but we don't have ranges in C++03. The available template parameters also |

steer programmers toward sensible restrictions. E.g. excluding minor civs from a loop but not Barbarians is suspicious, or counting a player's vassals but not its teammates.

Efficiency:

For `MAX_CIV_PLAYERS=18`, the BtS loops are actually very efficient, even if half of the players aren't used. I guess CPU branch prediction works very well for `isAlive` and other commonly used checks. Loops with small bodies also get partly unrolled by the compiler (into a `for i=0 to 3` loop with six repetitions in its body). However, when `MAX_CIV_PLAYERS` is increased, the BtS loops become somewhat inefficient when most players remain unused. That's one reason why 8-player games run more slowly with a 48-civ DLL than with an 18-civ DLL (another reason is that a higher `MAX_CIV_PLAYERS` value causes memory to be allocated for the unused players). Excluding agents that have never been alive from most of the loops is a step toward distributing only a single (31-civ) DLL. In any case, it's important that the agent iterators are efficient as agent loops occur very frequently throughout the code base.

I use two sets of enum values to specify predicates that the agents need to satisfy. The first says something about the status of the agent, e.g. whether it is alive, a major civ, a vassal; the second enum requires a particular relation with a second agent, e.g. being on the same team, not having the same master or having met. Bitmasks would be more flexible, allowing arbitrary logical operators, but I don't think that much flexibility is actually needed: The frequently needed predicates are mostly mutually exclusive. Bitmasks would also make a (highly) efficient implementation more difficult, and I don't want to use bitwise logic all the time in loop headers.

Not every combination of the two predicates should correspond to a cached sequence of agents. It might be fastest to cache most of them (hard to say), but implementing cache updates at the proper code locations is clearly not always worth the programming effort. Therefore, there's a third enum that lists the available cached sequences, e.g. `CIVS_ALIVE` (all non-Barbarians players and teams currently alive). The task of an agent iterator is to map the given predicate(s) to a cached sequence and to check any predicates not implied by the sequence on the fly; for example, if the predicate is `FREE_MAJOR_CIV` and the cached sequence `MAJOR_CIV`, then `isAVassal` needs to be checked for each agent in the sequence as the iterator advances. The iterator also hides the available cache data from the user.

Syntactically, I would have preferred to take the predicates as constructor arguments. I've implemented that too ([Git commit](#)), but it turned out to be a bit slower than I'd like. With the predicates as template parameters, much of the logic can be resolved at compile time. In particular, when a cached sequence matches the predicates exactly, the optimized assembly is essentially equivalent to an iteration over a vector.

I've run some performance tests (`AgentIteratorTest.cpp`) using the TSC profiler (see [003o](#)): For example, in a game with 8 players and up to 18 allowed, a loop over all major teams alive took 104 CPU cycles using BtS code, 106 cycles using an agent iterator and 96 cycles (arguably the fastest possible) directly using a vector. A loop over all players alive took 160 cycles with BtS code, 119 with an agent iterator and 102 with a (raw) vector. All members of a team: 124 (BtS), 68 (iterator), 52 (vector). For sequences that aren't (fully) cached, the iterator still performs a bit worse than the BtS code. Going through all free major teams alive took 384 cycles compared with 100 in BtS. One reason for this is that the iterator code can't be inline-expanded in this case (otherwise, header inclusions would get messed up). It's not really a significant difference though, I think, and if it turns out to be significant, more cached sequences can be added. When up to 48 civs are allowed and only 8 used, the iterators outperform the BtS code in most cases, e.g 167 vs. 406 cycles for players alive. I'm not sure why the maximal number of civs affects the performance of the iterators at all;

	must be some side-effect.
	As for the iterator interface, I've stuck to the C++ idiom of treating iterators as pointers, which is nicely compact, but I use a Java-style `hasNext` function rather than a comparison with some end-of-sequence constant.
See also	056 allows scenario files to be read that were created with a DLL allowing fewer civs than the current DLL.
	advc.enum reduces the memory allocated for unused players.
In the process of converting BtS/K-Mod loops to agent iterators, I've made minor functional changes that aren't always marked in the code. Most of these changes concern the treatment of Barbarians, minor civs, dead agents or unmet agents.	

advc.plotr	Iterators over plot ranges
See also	CityPlotIterator under advc.enum (because that one is coupled with the introduction of a "CityPlotTypes" enum).
	Comments in `PlotRadiusIterator.h`.
	C2C has a rectangle iterator: Git commit (At least for AI purposes, that doesn't seem like a very helpful generalization.)

AdvCiv	*BtS*

Example:
 	
``
for (SquareIter it(kCenter, iRange, false);
    it.hasNext(); ++it)
{
    CvPlot& kPlot = *it;
    // ...
}
``
 	
This will traverse the non-`NULL` tiles in a square of length 2*`iRange`+1 centered at a given tile or unit with the exception of the center itself (`bIncludeCenter=false`). The order of traversal is a spiral, i.e. tiles at a step distance of *i* are traversed before those at a step distance of *i*+1.
 	
There is also a "PlotCircleIter" that uses the same algorithm but skips tiles at the corners of the square, returning only tiles within a plot distance of at most `iRange`. (For `iRange=2`, this makes PlotCircleIter a less efficient implementation of CityPlotIter.)
 	
Almost all the BtS-style loops over square and (approximately) circular areas have been replaced with SquareIter, PlotCircleIter and CityPlotIter loops.
	BtS style:
	
	``
	for (int iDX = -iRange; iDX <= iRange; iDX++)
	{
	    for (int iDY = -iRange; iDY <= iRange; iDY++)
	    {
	        if (iDX == 0 && iDY == 0)
	            continue;
	        CvPlot* pPlot = plotXY(kCenter.getX(),
	                kCenter.getY(), iDX, iDY);
	        if (pPlot == NULL)
	            continue;
	        // …
	``
	
	The 0-check can take different forms, e.g. `pPlot == &kCenter`.
	
	Since the movement radius of a unit is a square ("step distance" metric), these loops are quite frequent in the Unit AI code. For unit movement, the center (current location of the unit) is usually disregarded.

Rationale	Readability mostly; the verbosity of the BtS code also makes it error-prone. (Other modders had fixed a couple of errors; I didn't find any new ones.)
	A more general "plot range" iterator might be neater, but there isn't much code duplication between SquareIter and CityPlotIter (the algorithms for generating the tile coordinates are completely different), and they were easier for me to write separately.
	The spiral pattern is intended to act as a sensible tie-breaker for argmax computations.

	It could, in principle, also speed those computations up, but that would require checks that cut the evaluation of a tile short when its value can no longer exceed the current maximum. For the most part, such checks don't exist, but perhaps they could be added in expensive tile evaluation loops. In terms of overhead, spiral traversal should not be appreciably slower than scanline traversal. (I haven't profiled it though; it might be that the memory layout of the CvPlot array favors a scanline.) Starting at the center has the advantage that the center tile can be skipped without any checks in subsequent iterations. In any case, having the square range traversal in a single place will make it easier to optimize if needs be.
Tbd.	A parameter for skipping tiles in other areas would be nice to have. Since the iterators return CvPlot references, it seems that `CvPlot&` is now more commonly used overall than `CvPlot*`. So, I think it's time to change function parameters to `CvPlot const&` or `CvPlot&` when possible. Iterator over the whole map. See *Tbd.* under advc.enum.

004	Minor usability improvements
See also	004a (see below): Changes to Bulb help 004b: Found-city button projects expenses; delete-unit button projects savings. 004c: UI changes concerning bombardment 004d: Leads-to info in promotion button help 004g: Misc. tweaks to game text 004h: Highlight full city radius when settler selected 004i: Announce amount of gold stolen by enemy spies 004k: Disable rarely used Sea Patrol mission by default 004l: Sentry Heal 004m: Default settings for camera distance, FoV, active map layer 004n: Show all garrisoned units on city screen through a single click 004o: No start-revolution choice on new-civic popup 004p: No commerce breakdown on city screen if slider at 0 004q: Show sum of diplo modifiers in leader hover text 004r: Changes to the announcement of resources revealed by newly acquired tech 004s: Changes to commerce and yield curves on Graphs tab [004t](004t): Option to disable click-on-map exitfrom city screen [004u](004u): More info in announcements of Great General deaths [004w](004w): Streamlining, omission of misc. help text that wasn't really helpful [004x](004x) cancels redundant minimized popups. [004y](004y): Full-screen Sevopedia and other tweaks to 'Pedia [004z](004z): Tweaks to map layers [101](101) adds help text about occupation and revolts to the Nationality bar (city screen). [120c](120c) allows hiding the espionage slider when it's at 0. [251](251) shows the start turn on the Settings tab when it isn't turn 0. [085](085) lets the scoreboard expand on mouse-over. [210](210) adds a few BUG-style alerts. [106](106): Various changes to on-screen announcements and the replay log [071](071) adds an option for notifications about first contacts (on-screen message or diplo popup) to the "Alerts" tab. [151](151) shows the previous religion or civic in messages about changed civics/ religions. [152](152) shows currently offered war trades on the Glance tab. [153](153): Automatically split up human unit groups at certain convenient times [075](075): Automatically wake units in cargo (at certain convenient times) [047](047) Tweaks to resource tile help text

	048: Combat odds help text (including some changes to ACO) 059 shows health/ happiness effects of features and improvements in tile hover text. 060 suppresses the add-to-spaceship popup. 061 makes the hover text for unit stacks on the main map more compact. 063 shows additional Actual Effects when Alt is held down. 064 adds BULL help text to the Whip button and allows hurry tick marks to be disabled. 064b adds overflow info to the production yield hover on the city screen. 065 removes the "Show Culture Turns" option; always shows culture turns. 106d changes BUG default settings for Civ4lerts and the scoreboard. kekm.30 adds civ and leader icons to the scoreboard (disabled by default). 066 frees up some space on the Foreign Advisor. 067 overhauls the BUG game clock. 068 lets changes to the BUG options for tech era coloring take effect without a restart. 069: Changes to the PLE component 070 replaces BUG's "Gold Rate Warning" option. 078 allows disabling BUG's GP bars until there is some progress toward a GP. 072 extends the show-deal-turns BUG option a bit. 073 Trade denial hovers from BULL; revises the layout of the "Resources" tab. 074 excludes some "must-be-joking" items from the Trade screen. 077 revises the "Demographics" tab (Info screen). 154: Unit cycling button 088: Key combination for unselecting all units 090: Longer FoV slider, more sensitive toward the right end. 091: Score graph sometimes shown when unable to see demographics 094: Production decay warnings from BULL 095: Option for wide city bars (not available in K-Mod, not optional in BUG) 096: Some tech tree hover text reflects the current game state 076 revises the player options menu (Ctrl+O). 002f: Changes to city bar icons 011b: Tile hover for partial worker builds 099f: Tile hover for culture on unowned tiles 190: Additions to the Settings tab (Victory screen) 910: Tech hover text shows speed-up from knowing multiple alternative requirements. 106m adjusts the dimensions of the replay screen to the screen resolution. 104m cancels expired peace treaties at the start of a round. 002l deals with sounds being played too many times at once. 172 removes culture rates from the Religion Advisor (because they no longer depend on the current state religion) 186 enhanced city bar hover text, mostly from BULL. 188: Tweaks to BUG anger timer on city screen 189: Starvation turns on the city bar 190: Additional info on the Settings tab (Victory screen) mnai: Detailed civic info for civic buttons on tech tree
Tbd.	BULL merge: Widescreen for Foreign Advisor; coloring of ratios on the Espionage screen (just use one coloring scheme; no options); option for building icons on city screen; some misc. city screen hovers; spaceship countdowns on Victory tab. Something like the BULL Members tab (Foreign Advisor) to help the player tally votes. Maybe HUD civics from BAT (requested here). Auto Save options: DLL code (just two lines) already merged; tagged with "BULL - AutoSave". Will need to add `gameStartSave` and `gameEndSave` to `CvAppInterface.py`. An option to make a second auto-save upon pressing "End Turn" would be nice to have. (But the BULL code doesn't do that.) Unit starting XP (tie that to the Alt key; no option needed); possibly tooltip for Drafting

from BUFFY. City Bar Tile Hover and/or City Tile Highlights (SVN revisions [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#))

Platy UI merge:
Smaller leader/civ icons on Military Advisor; Platy's Religion and Corporation Advisors (but keep the BUG Religion Advisor as an option); enhanced Statistics tab. Military Advisor should also be enlarged; [this](#) MNAI commit along with [these](#) (updated links: [1](#) [2](#) [3](#) [4](#) [5](#)) could be helpful; though I don't think I'll bother with the overblown BUG Military Advisor. MNAI also refactors the Customizable Domestic Advisor. Enlarging the BUG/Vanilla Religion Advisor isn't a priority if I'll merge Platy's version (which is already full-screen). Perhaps the Platy Religion Advisor should sort the cities by the selected religion (but not when only hovering over a religion button); related CFC [post](#). CFC post with some screenshots: [link](#)

From History Rewritten: [CFC post](#) by me
Espionage screen – beige panels, maybe reset weight button (requested [here](#)), go-to-city, investigate city buttons; Dawn of Man screen – see [704](#); Victory tab – leader icons, move Time victory to the bottom; I don't think I want to adopt anything from the city screen (despite a suggestion to that effect [here](#))

C2C has progress bars on the tech tree: [Git commit](#), [screenshot](#)
Probably too much work. Also, I think I'd like a wider bar across the tech name with a higher alpha value so that it stands out less and so that the tech boxes don't need to be enlarged.

Some mods have additional automation options. One that has been [requested](#) for AdvCiv (I don't think any mod has this): Automated workers never replace Towns and Villages and replace Forts only on workable tiles.

Another automation [request](#): interface mode that allows protecting individual tiles from automated workers. The implementation should store the protected tiles as a serialized `vector<CvPlot const*>` at CvPlayer, add a function `isAutomationProtected` like the `CvPlayer::isAutomationSafe` function I've already added (call locations should also overlap; also check uses of `PLAYEROPTION_LEAVE_FORESTS`) , add a new interface mode akin to Sign mode that can either add or remove protection from a plot. Hotkey should be Alt+P. `CvGame::updateColoredPlots` should apply some light hue of blue, red or white, not sure in which `PlotStyle`. The cursor should ideally use the same style with higher opacity. A loading screen hint and entry in the Pedia shortcut list should be enough for visibility. Don't want players to go and "look for uses" for this feature – there probably aren't many.

Misc:
Show the turns-left countdown only when at most 30 turns away from time victory (instead of 100); inspired by [RFCEurope](#).

Announce when a tech that grants a free Great Person is first discovered. Credit: Civ 4 Reimagined [1.4](#).

The choose-production popup always includes the city name (BtS says "in this city" sometimes).

The UI never shows damaged units at full strength nor at 0.0 strength. E.g. a Warrior with 98/100 hitpoints is shown with strength 1.9/2 instead of 2.0/2 (exact value: 1.96/2).

Overhaul of the BUG menu: headings, help text, layout, color palette, default settings. Changed some colors of the "Detailed Food Info" option through `BUG_CIV4GameText.xml`. Mainly [this](#) Git commit.

Merged some misc. hover text from BULL: Code tagged with "BULL - Trade Hover", "BULL-Finance Advisor", "BULL - Leaderhead Relations", "BULL - Food Rate Hover". [086](#) and [087](#)

streamline that text a bit.

Help text for units that cause collateral damage says how many defenders can be affected.

Upon closing the Military Advisor screen, the game forgets which players were selected. (BtS: Remembers which players were selected and selects the active human player in addition when re-opening the screen.)

The non-customizable Domestic Advisor screen shows the strength (bicep) icon instead of the defense (tower) icon above the column that shows the number of military units stationed at a city. And the maintenance column shows costs including inflation (as on the city screen).

Update the status of the city unhappiness indicator on the main map in CvCity::changeMilitaryHappinessUnits, i.e. when unit moves into or out of a city.

Show effects of routes (Railroad) in hover text of improvements. (`CvGameTextMgr::setImprovementHelp`)

Make sure that an announcement is shown when the active player pillages an improvement (by ensuring in `CvUnit::pillage` that the pillage gold is at least 1).

The "BUG Statistics" tab is no longer optional – all it does is add statistics about terrain improvements; those are now always shown. And all the statistics are sorted alphabetically when the screen is loaded (rather than sorting by internal IDs).

When pairing cities for trade routes (`CvCity::updateTradeRoutes`), count trade route profit at times-100 precision. *Rationale*: Should lead to fewer ties (which are currently broken arbitrarily based on player and city ids) and should make it more likely that foreign trade routes are chosen over domestic ones. *See also*: It can be confusing when only domestic trade routes are shown despite available foreign partner cities as in this CFC thread.

Disabled the global-rank columns of the Customizable Domestic Advisor because they leak information. Show "?" in the coordinate columns until the world map has been centered. *Credits*: Inspired by the "Close to Home" mod (Git commit).

~~Hide gold from trade inventories when the one side has already put gold on the trade table.~~ Disabled this again in order to keep the amount of available gold on display; the code is commented out in `CvPlayer::updateTradeList`.

004a	UI support for the Discover ("bulb") ability
AdvCiv	*K-Mod/BUG*
The help text for the bulb button of a GP unit shows all techs that the GP will be able to discover if an additional tech is researched, e.g. "next tech: Astronomy (with Printing Press)".	The help text only says which tech the GP can discover right now.
GP Research on the Tech Advisor disabled by default.	BUG's "GP Research" option shows bulb paths on the Tech Advisor. Enabled by default.

Rationale	For players who don't plan their bulbs (long) in advance, the added help text should usually suffice. I'd like to show only the tech tree on the Tech Advisor.
Config	GP Research can be enabled on the Advisors tab of the in-game BUG menu.

If GP Research is enabled, info about techs that a GP could currently discover is shown in the footer area of the Tech Advisor.	Shown to the left of the tech tree, overlapping with the first column of techs. This looks like a bug; the whole tech tree is probably supposed to move to the right a bit.

Credits	Idea from RFC: Dawn of Civilization; I've also seen it in Platy's Tech Screen, so perhaps it's his work originally.
Rationale	Want to give the tech tree as much room as possible. Some Advisor screens use the

	footer for tabs, but, since the Tech Advisor has no tabs, it's unused space.	
For each type of GP, the following info is shown if applicable: • Preference order: All techs that the GP could possibly discover, sorted by priority. • Current tech: Tech that the GP would currently be able to discover. • Missing requirements: Techs that have a higher priority than the current tech and don't require the current tech. These are techs that will replace the current tech if their prereqs are researched. • After research: The tech that the GP would be able to discover if all techs currently queued for research were already discovered. Improved the help text a bit, though it's still not easy to understand I think.	• (same) • (same) • Techs with lower priority than the current tech for which all tech requirements are met. These are techs that the GP could discover if the requirements for the current tech weren't met. • (same) • Techs with lower priority than the current tech for which all tech requirements will be met after finishing all queued research. Difficult to guess from the help text (and layout) how all this works. E.g. the "after research" techs are just called "Future Technologies".	

Rationale	For optimal bulb paths, it can be necessary to avoid certain techs ([example](#)). I'm guessing that's why the BUG mod shows the lower-priority techs. Or the idea was to show which tech the GP would get if its current tech was discovered, but that's not always true (because the current tech could unlock another high-priority tech). So the BUG info is complicated and misleading. I don't think tech avoidance matters very often, and players that use this tactic probably have their bulb paths memorized or know how to figure them out from the preference order. The missing-requirements list should make the player aware of techs that the GP will be able to discover if one more tech is researched through other means. Example: If the current tech for a Great Scientist (GSc) is Compass, the player can switch his/her research to Compass to find out that the GSc will be able to discover Aesthetics once Compass is finished. This works the same way in BUG as in AdvCiv. But how to tell in BUG that the GSc could also discover Philosophy if Meditation was researched? AdvCiv lists Philosophy (and Paper) under "requirements missing".
Config	Implemented mostly in `CvTechChooser.py`.

AdvCiv	*BtS*
No "can be researched by a (e.g.) Great Scientist" help text on technologies.	Help text on a technology says whether that technology could currently be discovered by a particular GP.

Rationale	If the player does have such a GP, then the player can consult the Discover button of the GP to find out which techs it can discover. If the player doesn't have the GP, then, by the time a GP is born, the information may no longer be accurate. Most of the time, the help text is just distracting. If the player needs to know about bulb paths, he/she should consult the BUG Tech Advisor.
004b	Cost projections in action button help text

AdvCiv	*BtS*
The help text for the found ("Build City") button shows how much the Total Expenses (Economics	No such projection. Can only quicksave and found to see how costs will increase.

Advisor) will increase if a city is founded on the current tile. Does *not* project the gold income of the new city, e.g. from trade routes. The increase is computed as inflation times + maintenance for the new city, including State Property, vassal cities etc. + increased maintenance in other cities, including those temporarily exempt because of disorder or celebrations + increased civic upkeep + decrease in unit cost (from the +1 population) + decrease in unit supply and cost from the lost Settler	

Tbd.	Should also show free initial buildings (Palace, more with Medieval start or later) and the number of trade routes (predicting the yield would be tough).

The help text on the found button shows health from features, traits (Expansive) and freshwater. All features revealed in the (full) city radius count. While a Settler is selected, the help text of any tile shows health effects. Found-button help text shows the projected city tile yield.	Forests provide +0.5 health, Jungles -0.25, Flood Plains -0.4 and Fallout -0.5. City tile yield only shown once a city has been founded. Usually just 2 food, 1 production, 1 commerce but occasionally more.

See also	016 changes the effect of extra yields from random events on city tiles. 004h shows the full city radius when a Settler is selected. 059 show health and happiness effects from tiles surrounding actual cities.

Help text of the delete-unit button shows whether the deletion will decrease expenses for units: supply, unit cost (incl. Pacifism) and (for mod-mods) extra cost. If deleting the selecting unit(s) will not decrease expenses, but deleting additional units would, the help text says how many additional units would have to be deleted.

Tbd.	If the Finance tab (Economics Advisor) is checked before and after deleting a unit, the difference in unit expenses doesn't always agree with the difference predicted by my code. This is because the subtotals on the Finance tab include inflation, which isn't how `CvPlayer::doGold` actually computes expenses. Cf. comment in `EconomicsAdvisor.py`. Also, during anarchy, some of the expense items shown in help text on the Finance tab are positive. Should all be zero.

004c	Changes to bombardment, air bomb missions [not strictly UI changes]

See also	004g shows messages about bombardment immediately.

AdvCiv	*BtS*
Can bombard at 0 defense in order to prevent city defense from recovering. Air units and siege units with the ability to ignore defensive buildings have their bomb(ard) rate increased to match the building defense. For example, when a city that has 100% defense from a Castle but just 60% from culture, is bombarded by a Cannon (bombard rate 12), city defense is reduced by 20 percentage points. Units that ignore defensive buildings see 48% defense, the others 80%.	Can't bombard cities with 0 defense. If a city isn't bombarded for a turn, its defense begins to recover on the next turn. The ability of Wall and Castle says "+... defense (except vs. Gunpowder-based units)" and those Gunpowder-based units have the XML tag `IgnoreBuildingDefense`. However, Siege units with the tag don't actually ignore defensive buildings; they only ignore the bombardment reduction effect that Wall and Castle also have. In the example on the left, the Cannon player would see 60% before bombardment, and 52% after.

Rationale	I wouldn't mind Walls and Castles having defensive abilities against post-Medieval units, but the implementation is unacceptable from a UI pov. By the Modern era, stacks often consist entirely of units that ignore building defense, and then the player only gets to see the defense from culture, which somehow doesn't decrease as fast as it should. An alternative solution would be to show city defense including buildings when a siege unit is selected; in the example, the player would then see defense decrease from 100% to 88%. Not trivial to implement (can't just take away `IgnoreBuildingDefense` because siege units still need to ignore bombardment reduction), and my solution is conceptually simpler: post-Medieval siege units entirely ignore defensive buildings.

When applying bombardment damage (computed as described above), the city's defense damage percentage is set so that the city's defense modifier decreases by an amount exactly equal to the bombardment damage. In formulas: The damage percentage d_1 after applying bombardment damage b should be chosen such that the new defense modifier m_1 is b less than the present defense modifier m_0. As in BtS, m_0 and m_1 are computed from the damage percentages d_0 and d_1 and the city's maximal defense t (defense modifier when undamaged): $m_1 = floor((100 - d_1) * t / 100)$ From this, one can derive the following equation for d_1: $d_1 = 100 - ceil((m_0 - b) * 100 / t)$	Defense damage is stored as an integer percentage, meaning that the bombardment damage needs to be converted into a percentage and rounded. In BtS, the rounding can lead to a counterintuitive loss of 1 bombardment damage. Example: A city with a 60% defense modifier gets bombarded by 5 Catapults. 8 bombardment damage is taken times 100/60 and rounded down in order to convert the damage into a percentage; that's 13% per Catapult. After 4 shots, the city has 52% damage, i.e. is 48% undamaged. To compute the defense modifier, the 48% are taken times 60 and the result is rounded down: 28%. That's as expected: 60-4*8=28. The fifth shot, however, results in 65% damage and 35%*60% rounds down to 21% – the defense modifier has only decreased by 7.

Tbd.	Simply store the defense damage as the sum of the bombardment damage? This might lead to unexpected results in some circumstances, but I can't think of any. E.g. if a city constructs Walls during a siege, then subtracting the bombardment damage so far from the increased maximal defense would be pretty intuitive.

Help text for Walls and Castle says that defense is "raised to" 50% and 100% respectively. Defense is represented by the tower icon.	Says "+50% defense" for both.

Rationale	"+50%" is misleading because building defense isn't cumulative with culture defense.
Config	Implemented through a new XML tag "RaiseDefense". Walls have RaiseDefense 50 and Castle 100, i.e. this new tag is non-cumulative when it comes to building defense. The old "DefenseModifier" tag still works but is unused.

Don't count defensive abilities in AI evaluation of building obsoletion (looks like K-Mod did count them).	

Help text of Bombard button and in Air Bomb mode says by how much the defense modifier will decrease.	Generic help text saying that bombardment decreases the defense modifier. No help at all while hovering in Air Bomb mode.

Rationale	Not obvious when buildings and gunpowder units are involved (see above).
Tbd.	A breakdown would be nice – showing reduction from buildings, increase from ignoring building defense.

When hovering over an enemy improvement in Air Bomb mode, the probability of destroying that improvement through an Air Bomb mission is	The probability is shown nowhere (and isn't even explicitly computed). Not even the values that enter into the probability (current air bomb rate,

shown.	improvement's air bomb defense) are shown anywhere.
Tbd.	Should Air Bomb missions be allowed to target routes? (Only improvements currently.)
All missions that can be intercepted (air bomb, air strike, paradrop) show the best visible enemy interceptor and the interception probability while hovering for the mission target or for combat odds (right mouse button or Alt+hover). Except when that probability is 0. When multiple units are selected, also show which unit will get intercepted (i.e. which unit will execute its mission first).	No such probability display. Can only scan the nearby tiles for units with an intercept chance. Unit help text shows that chance, correctly adjusted to hitpoints in the case of damaged fighter aircraft.

Rationale	A player generally can't be certain about the interception probability because there could be a Fighter somewhere in the fog of war up to 6 tiles away from the target (jet Fighter: 10 tiles). Still, showing the probability of visible units seems much better than nothing.
	Could make a case for showing combat odds in Air Strike mode, i.e. the same info as Alt+hover (no odds really, but the air strength of the attacker and the combat strength of the non-air defender). But I think it would be too much text if non-combat help text (e.g. terrain info) is shown in addition. Currently, the player can choose between detailed combat info (Alt+hover or right mouse drag) and civilian info plus interception (Air Strike mode). Just too bad that the latter is what players intuitively use.
Tbd.	The next step would be to show odds for air combat. Big task.
	Maybe the first unit to execute its mission (assuming that multiple are selected) should be shown even when there is no interceptor. I'm just not sure exactly how to phrase the help text.
See also	[128](#) uses the non-cheating interceptor check (written for UI purposes) in AI code. [650](#) shows the interception chance in Nuke Mode.

Human group missions are halted when a unit gets intercepted unless the Stack Attack option is enabled. Exception: Group paradrops are only halted when a paradropping unit gets destroyed by an interceptor or the first time that an interception occurs from the fog of war.	When a group is ordered to air bomb, air strike or paradrop at a tile, all units in the group execute the mission if they're able to, regardless of interceptions.

Rationale	This seems pretty similar to group combat on land and water – the game should assume that the player wants to reconsider his or her actions after each interception. Air combat isn't as lethal than land and water combat, and insofar perhaps less unpredictable, but, on the other hand, interceptors can intervene from the fog of war.
	Exception for paradrops: A single enemy Fighter can intercept any number of paradrops without taking damage. It seems too tedious to move a whole stack of Paratroopers one by one then.

Custom icon for the Air Bomb button, superimposing the Bombard icon on the original Air Bomb icon.	The Air Bomb icon looks very similar to the Air Strike icon. Even though the buttons are right next to each other, they look virtually the same.

Rationale	The Bombard icon (red-white targeting circles) is very distinctive, and Air Bomb against cities works almost exactly like Bombard. It's still not so clear that this is also the button for bombarding improvements. Maybe that really should've been a separate mission, but I'm not going to experiment with that. I also don't think that superimposing the Pillage icon in addition to the Bombard icon would look good.

Config	BtS uses an icon atlas for the Air Bomb button. Since I didn't want to include a modified atlas in the mod, I've changed the button icon path to an individual graphic instead. See comment in `Civ4MissionInfos.xml`.

Unit help text for air units shows the current air bomb rate.	Civilopedia shows the maximal air bomb rate of a unit; unit help text doesn't show any air bomb info.

Rationale	Most players may not even be aware that the air bomb rate (unlike the bombardment rate) is decreased when a unit is damaged.

AdvCiv	*K-Mod*
Closed some fog-of-war information leaks caused by the Air Bomb mission: Tile ownership, unrevealed cities, removed improvements, depleted city defense. Ordered Air Bomb missions are no longer guaranteed to find a target; can e.g. happen that the targeted improvement no longer exists. In that case, the recon effect is the only result of the mission, and a special announcement is shown.	K-Mod already takes care of somemore egregious leaks. Still guarantees that any Air Bomb mission that can be ordered will also find a target.

See also	[001i](#) deals with leaked info about routes in the fog of war.

AdvCiv	*BtS*
When a group of (human or AI) units is ordered to bombard a city, then the missions are executed (until city defense reaches 0) based on a priority function that takes into account bombardment rate (the higher the rate, the higher the priority, but avoid overshooting), city attack odds, collateral damage (try not to bombard with effective city attackers) and promotions (only for human units). When the city is expected to fall easily, then damaged units (despite having poor attack odds) are deprioritized – so that they can heal earlier.	The missions are executed in the internal order of the group. I'm not sure what that order is, but it seems pretty unpredictable.

Credits	Requested by Leoreth ([CFC post](#); see 3rd item), and he also provided concrete ideas for the priority function a few posts below.

Rationale	Requested as a UI feature, but I think there is a tangible benefit for the AI as well; AI bombardment missions are generally executed by stacks, not individual units.

See also	If not all selected units are needed to bombard a city down to 0 defense, then [153](#) splits off units that didn't get to bombard into a separate group and selects that group.

When a group is ordered to air bomb a city, then the missions are executed (until city defense reaches 0) in an order that prioritizes units with high defense damage, tries to avoid overshooting and (more of a tiebreaker) deprioritizes valuable units (high production cost, XP).	Executed in the internal order of the group.

Rationale	I had written this simpler heuristic for both air bomb and bombardment missions. I don't think it was quite up to snuff for bombardment (see Leoreth's criticism in the thread linked above), but it may still be good enough for air bombing. At least it's predictable. The new bombardment code isn't suitable for air bombing.

AdvCiv	*K-Mod*

When a group of units is ordered to air bomb an improvement, then the missions are executed (until the improvement is destroyed) in an order that prioritizes the current air bomb rate (and thus the chances of success). As a tiebreaker, production cost and XP are used (prefer using low-value units for bombing). Likewise, when a group of units is ordered to pillage an improvement, production cost and XP are used as tiebreakers.	The internal order of the group is used for air bomb missions, but, for pillaging, fast-moving units are preferred. (BBAI had disabled group pillaging entirely; K-Mod re-enabled it.)
Rationale	Important for the displayed chance of success (see blue box higher up). Only one probability is displayed, and the player will probably assume that it's the best probability. And a player will probably also assume that the best unit will carry out the mission (just like the best attacker comes forward in a group attack) – and not some arbitrary unit. (Pillaging doesn't really belong under this change id, but the logic is quite similar to air bombing.)
AdvCiv	*K-Mod*
When cycling between units, try to avoid cycling between air units (or nukes) and civilian units.	Air units and civilians are treated as the same category of units.
Rationale	Seems like an oversight. Assuming that `CvUnit::canFight` identifies military units is a somewhat common mistake. (It's actually just units that can engage in regular combat.)
Config	K-Mod handles the unit cycling order through `CvSelectionGroup:: groupCycleDistance`
Credits	CFC user MightyToad made me aware of the problem in this post ("side note" above the 2nd quote box).
Tbd.	Not sure how much I've improved the situation. I haven't implemented the suggestion to always select all aircraft (around the world) first. Perhaps that should be a BUG option.
004d	AI says "not right now" to peace when war is recent
AdvCiv	*K-Mod*
When a player tries to broker peace, the Trade Screen says "not right now ..." if the war is still too recent. The AI will agree to brokered peace with a third civ as part of a peace treaty with the human even if the AI would normally still refuse to talk to that third civ.	Says "we'd love to, but you'd have to ask them" instead. Often, both sides say this. (As far as I recall, peace could be brokered even when a war was recent in BtS.)
004e	Leads-to info added to promotion buttons
AdvCiv	*BtS*
The help text for promotion buttons lists the promotions enabled by the current promotion. E.g. "Flanking I Leads to Flanking II, Navigation, Sentry"	Can only look this up in Civilopedia.
004f	Disabled celebrations

No celebrations (We-Love-The-King Day) ever.	If a city has at least 8 population, no anger and no bad health, celebrations happen with a probability of population / 1000. The reward is 0 maintenance for 1 turn.
Rationale	Celebrations grant just -3% maintenance on average in a size 30 city, which is insignificant. I'm not even listing this as a balance change. Celebrations were potent in Civ 2 but already irrelevant and confusing in Civ 3. Something no one will miss who doesn't know it's disabled.
Config	`WE_LOVE_THE_KING_POPULATION_MIN_POPULATION` in `GlobalDefines_advc.xml`
004g	Misc. changes to confusing help text (not a complete list)
See also	062: AI diplo comment when canceling a vassal agreement

"Our shared borders spark tensions"	"Our close borders spark tensions" Too similar to "closed", and doesn't imply that they're touching.
"We don't like you enough"	"We just don't like you enough" Sounds like this is the only obstacle – but often isn't.
"Your x has attacked a y: 22% damage." Removed the minus sign.	"Your x has attacked a y: -22% damage"
"Reduced city defenses" message shows the defense percentage without Walls/ Castle if the bombarding unit ignores building defense.	Always shows the defense percentage including buildings.
"Some wonders on this continent are making us happy" for the Notre Dame ability. (Only changed in English and German)	"Some buildings are making us happy", listed twice when there is also e.g. a Colosseum.
"The anarchy is over" in white letters	In red letters, like it's a bad thing.
"We are afraid of their military might" when sponsored war refused on account of nukes	"Surely, you must be joking."
	Both polar desert and shelf ice are called "Ice".
Polar desert called "Snow" instead of "Ice".	"disrupts the trade route so that no intercontinental trade can pass through the blockaded tile"
Help text of Blockade missions says that only enemy trade is affected.	Listed as "free city commerce". (With translations in the vein of "free-flowing commerce.") Expansive: "from Civilization"
List culture from Creative trait in city culture breakdown as "from Traits". List health from Expansive trait as "from Traits". *Tbd.*: "Yeah" for happiness from Charismatic trait is too obscure (and pretty dumb). Can't use "from Trait" there though; has to be a sound bite.	
Refer to bonus resources as "Resources" in health help text.	Seems that "resources" is used predominantly, e.g. "we enjoy our luxurious resources", but, for health it says "Bonuses". Positive modifiers are also referred to as "bonuses" sometimes.
Announce bombardment of human cities immediately (`bForce=true`) and include the name of the bombarding unit (inspired by MNAI). Announce pillaging of human improvements immediately as well.	

Replace the (slang) term "plot" with "Tile" in some game text. (Didn't bother with this for 'Pedia, WorldBuilder, map scripts and random events.)	

004h	Highlight full city radius when Settler selected; no yield icons

When a Settler is selected, the full city radius is highlighted.	Only the inner radius, i.e. the adjacent tiles are highlighted.
No yield icons are shown (unless they're always shown through "Display Yields"). (To implement this, `CvInterface::toggleYield VisibleMode` was no help. I had to replace `CvUnit::isFound` so that the EXE doesn't even know whether a Settler is selected.) Nothing is highlighted if any non-Settler unit is selected along with a Settler.	Even if "Display Yields" (button above the mini-map) is disabled, yield icons are shown when a Settler is selected. The highlighting is based on whether at least one Settler is selected (`CvInterface:: canSelectionListFound`).

Rationale	Highlighting the adjacent tiles isn't helpful at all; it's not even a reliable projection of the borders right after founding because some of the highlighted tiles could already be owned by another civ.
	The yield icons are useless to me personally, but I know some players always keep them enabled, perhaps, in part, because icons on worked tiles are highlighted, making it easier to tell which tiles are being worked. I'm not sure if anyone wants to see the icons only when a Settler is selected – I normally choose my city sites before even training a Settler.
Config	Two options on the "Map" tab of the BUG menu
Tbd.	When settling on a tile marked with a dot on the Strategy layer, it would be nice to remove that marker automatically.
See also	004b shows info about city maintenance and health when a Settler is selected. 009c (removal of Map Finder) makes room for the options on the BUG menu.

004i	Message about gold stolen by enemy spies says how much was stolen
Credits	Idea from Civ 4 Reimagined 1.2
See also	120d: The Espionage screen shows the amount of gold to be stolen in advance.

004j	Changes to "Regenerate Map"

AdvCiv	*BtS*
~~In singleplayer games, the map can be regenerated until (incl.) turn 3. Exception: Can't regenerate after any civ has met another civ.~~	Can only regenerate the map on turn 0 (unless civs meet each other already on turn 0).

Rationale	More convenient for testing, but, on second thought, regeneration past turn 0 might not be entirely clean. Regeneration does not reset the game, player and team objects; just destroys all units and cities. Also, it's not such an inconvenience to reload the initial autosave.
Tbd.	Perhaps a full reset wouldn't be so hard to implement.
	At least I should try to get rid of the have-met exception on turn 0. Resetting the has-met flags is simple. Years-of-peace and perhaps other AI memory starts to count when another civ is met, but that's not a problem on turn 0, and resetting memory counts is also easy enough. A human civ could immediately declare war, so I at-war flags would have to be reset. When starting in a later era though, various trades could be made on turn 0 ...

After regenerating the map, the Dawn of Man (DoM) screen is shown again, and the camera centers on the player's active unit.		The DoM screen isn't shown after regenerating, and the game tries to center the camera on the player's starting plot, but somehow this has no effect; camera stays at the previous starting plot.
Config	Optional through `GlobalDefines_advc.xml`. If disabled, the camera still centers correctly (through change 003r).	
Rationale	Without the DoM screen, the player sees the old starting location slowly fade to black. I'm not sure if that reveals parts of the new map that the player isn't supposed to see, but, even if it doesn't, it's confusing. The DoM mostly obscures that.	
See also	704: Other changes to the Dawn of Man screen. 001: Workaround for a problem with the camera position after loading a savegame in which the active player has no units.	
Extra yields added through Python are reset before regenerating the map.		Only random events set extra yields. Can't occur on the initial game turn, so no need to reset extra yields.
Rationale	For mod-mods that might set extra yields during map generation.	
See also	016 makes an exception for extra yields so that peaks can be made workable through Python.	
004k	Disable the Sea Patrol mission for both human and AI players.	
Rationale	This mission (added by the BtS expansion) has a very narrow effect; few players seem to ever use it and many may not even know what it does.	
Config	Can re-enable the mission on the "General" tab of the BUG menu.	
See also	Explanation on CFC. To clarify: Sea Patrol forces the pillaging unit to attack, so units on Sea Patrol do receive defensive combat modifiers. Also, the AI does use Sea Patrol; it's just unlikely to ever matter.	

004l	Fortify-Heal works like Sentry-Heal outside cities	
AdvCiv		*BtS*
When a unit is fortified "until healed" outside a city, that unit will ask for orders when a hostile unit approaches. Forts behave like cities in this context, i.e. units healing in Forts don't wake up until fully healed. Added a new mission `SENTRY_HEAL`, but that's only a technicality so that the help text can say "fortify until healed" when in a city and "sentry until healed" otherwise.		Units that are fortified until healed only ask for orders once they're healed.
Rationale	BULL has a separate Sentry-Heal mission that is missing from K-Mod. My change to Fortify-Heal should result in the desired behavior in most cases. Units already have so many rarely used buttons; don't want to add another. BAT also uses a single button (and has a menu option for switching to the BtS behavior). I've tried using the Sentry-Heal graphic from BUFFY (a monocular on top of a canteen) for the Heal button when outside a city, but I think this draws too much attention to what is supposed to be a low-key change.	
Units on Sentry (and units healing in a non-city tiles) are not woken up by enemy units that start their move already within visibility range.		A unit on Sentry is woken up if and only if there is an enemy unit within the Sentry unit's visibility range at the start of the turn. How the enemy unit

	got there doesn't matter.
Rationale	Sometimes, enemy units keep stalking around a Sentry unit because they're too weak to attack it or entirely unable, e.g. a ship vs. a land unit. Now the Sentry is only woken up once in such a case – unless the enemy keeps moving in an out of visibility range. It wouldn't be difficult to keep track of encountered enemies indefinitely (regardless of whether they move out of range), but the player may actually want to be alerted when an enemy unit returns, and a player isn't really supposed to know whether a unit that has moved out of sight has returned or whether it's a different unit with the same characteristics.
	The BUFFY mod has an optional "Sentry (Land)" mission to address problems with land Sentries being woken up by ships. That's a much clunkier solution, and not really suitable for AdvCiv because, in AdvCiv, Barbarian ships may well carry land units (see [300]).
See also	Depends on [003k] (workaround for adding data to CvSelectionGroup). [075] wakes embarked units up when a ship reaches land with its last movement point.

004m	Default camera distance, field of view (FoV), layer icons

AdvCiv	*K-Mod*
Decreased the default FoV value; with only a slight adjustment to screen dimensions. FoV slider still hidden by default. Added a brief explanation of the field-of-view setting to the hover text of the option on the BUG menu.	BUG introduced the FoV slider; previously, the value was only adjustable through XML. K-Mod hides the slider by default and adjusts the default FoV value based on the screen dimensions: "*bigger FoW for bigger monitors*"

Config	The slider can be enabled on the "Map" tab of the BUG menu. If "Remember Field of View" is checked, the value on the slider is stored in `My Games\Beyond the Sword\ AdvCiv\Settings\BUG Main Interface.ini`. "Field of View" can then be unchecked again in order to hide the slider.
Rationale	The lower the value, the smaller the field of view, i.e. fewer tiles are shown on the main interface and the city screen. If fewer tiles are shown, one has to zoom out closer to the Globe view boundary to get an overview. This, in turn, means that the camera angle becomes more top-down, which is what I want. A larger screen can show a larger field of view, however, that will make the resource bubbles (if enabled) appear very large. They scale (only) with the screen resolution (cf. *Tbd*. under [004z]).
See also	BUG developers discussing the creation of the field-of-view slider: [link] [090] makes some changes to the field-of-view slider.

Increased the height of the Globe view boundary a bit.	
Rationale	Somehow this yields a slightly steeper angle that I can't seem to achieve just through the FoV setting. Or maybe the FoV would have to be so small that the workable city radius would no longer fit on the city screen.
Config	`XML\Misc\Civ4DetailManager.xml`
See also	CFC [post] by Toffer90 about the settings in `Civ4DetailManager.xml`. [002h] disables the passing clouds.
Tbd.	I'd like to increase the height of the Globe view boundary further. Currently, I play near that boundary so that the camera pitch is almost top-down; but it would be nice to be able to zoom out a bit farther from there without entering Globe view. The camera pitch

	can be set arbitrarily through `CyCamera.setBasePitch`; see the [BlueMod](#) for example. It might be possible to intercept the mouse wheel input and set a steeper pitch. (Letting players freely adjust the pitch won't work well because the mouse wheel will zoom beyond a perpendicular angle when the base pitch is steeper than the BtS default.)

AdvCiv	*BtS*
Let the DLL set the `CAMERA_START_DISTANCE` value based on the FoV value and era.	Set statically in GlobalDefines. Determines the default camera distance, but also affects music volume and zoom granularity.

Rationale	The default zoom is way too close and flat, especially when loading a savegame – except, perhaps, when playing with a high FoV value; it seems that players who use a high FoV value like playing at a flat angle.
Config	Option on the Map tab of the BUG menu. By default, the camera start distance is set automatically as described above, but players can also configure a fixed value.
Tbd.	Perhaps a fixed value should be the default. Players keep running into this problem (see links below); a fixed value might be less likely to not work well at all for some players. On the other hand, more players could then end up with a value that works only somewhat; perhaps better to make players aware. (I've also added a loading screen hint.) It's also not clear what a good fixed value would be; the BtS value isn't going to work well because, by default, the mod doesn't use the same FoV value as BtS does. Better to use city count instead of era (while still updating the `CAMERA_START_DISTANCE` only upon entering a new era)?
See also	Hover text of the BUG option. Bug reports about music being inaudible at high FoV values in AdvCiv 0.97 and earlier due to a high `CAMERA_START_DISTANCE` set in XML. [1](#) [2](#) CFC posts suggesting that a camera start distance based on FoV isn't suitable for all players. [1](#) [2](#) [3](#)

Enable the Resources layer ("bubbles") at the start of a new game.	No layer is enabled at game start.

Rationale	Most players seem to play with the Resource layer.
Config	Can be toggled on the "Map" tab of the BUG menu.

The currently active layer is stored in savegames. If the saved layer requires Globe view, then no layer is enabled after loading a savegame. To implement this, I've added Python code that tells the DLL when a layer is enabled or disabled.	After loading a savegame, the same layers remain active as before loading. When loading from the main menu, no layer is enabled. No function in the DLL interface for checking the currently active layer.

Rationale	The layer at the time that the savegame is created is more likely to be suitable for the saved game state than the layer that is active when loading.
See also	A DLL function for checking the current layer was needed anyway for [003d](#) and [102](#).

004n	Arrow button next to unit icons on city screen shows all local units

AdvCiv	*BtS*
One normal click on the right-arrow button on the city screen shows icons for all local units.	One full row of unit icons is shown when opening the city screen. If there are more units in the city tile than fit in one row, each click on the right

		arrow next to the unit icons shows one additional unit. If Ctrl is held when clicking, 10 more units are shown.
Rationale	At least at 1280x1024 resolution – and I think most players have that much –, even 100 units don't really obstruct the city screen. Showing them one by one is needlessly slow and complicated.	

004o	No start-revolution button on the new-civic popup	
AdvCiv		*BtS*
The change-civic popup only has the buttons "No, the old ways are best" and "Let's see the big picture." Exception: When there is only one civic that the player can change to, the start-revolution button is shown in addition. This usually happens when the player discovers Bronze Working.		"Let's get this revolution started" directly starts a revolution.
Rationale	Players sometimes forget other civic changes that could be made along with the one suggested by the game (especially if more than one civic becomes available at a time). This is frustrating and happens easily.	

004p	No commerce breakdown on city screen if slider at 0	
AdvCiv		*BtS*
When showing help text for any of the four commerce types on the city screen, a breakdown of raw commerce is shown only when the slider position is above 0.		Raw commerce is shown even when the slider is 0. E.g. "Culture: 1000 ==== Base commerce: 50 +50% for Capital ------- 0% of 75 = 0"
Rationale	Makes the culture-bar help text harder to read, and could be confusing for new or returning players since the culture slider isn't even available at the start of a game.	
Don't show the player's total culture output on the main interface.		Total culture output (sum of the culture rates in all cities) shown next to the culture slider.
Rationale	Not an important statistic; not worth being constantly on display. Even a bit confusing as total culture doesn't have any effect of its own.	
Config	Option on the BUG menu (General tab). As requested [here](#) (CFC).	
See also	[120c](#) hides the espionage slider when it's at 0	

004q	Display of relations modifiers	
See also	This would make less sense if some modifiers were hidden; but [sha](#) reveals them all.	
AdvCiv		*BtS*
Sum of the relations modifiers shown along with diplo breakdown. For example: Pleased **(+7)** towards Gandhi +1: "Years of peace …		Players have to sum up the modifiers by themselves to find out the total relations value: Pleased towards Gandhi +1: "Years of peace ...
If not all modifiers are revealed (randomized personalities), the sum of the revealed modifiers is displayed.		BUG shows the sums on the Glance tab. That's actually based on searching the attitude string generated by the DLL for plus and minus signs.

	Therefore also only counts revealed modifiers.
See also	sha reveals all relations modifiers except when playing with randomized personalities.
When playing with randomized AI personalities, unusually high memory-based relations modifiers are (incorrectly) displayed as just +1 or -1. (Until AdvCiv 0.96d, all memory-based modifiers were – accidentally – hidden when playing with randomized personalities.)	E.g. "-2: You declared war on our friends" when only only one friend has been attacked unequivocally gives away Gandhi's personality.
Tbd.	The types of AI memory to which this change applies are currently hardcoded in `CvGameTextMgr.cpp`. Would be better to check how many personalities deviate from the default and decide based on that whether to show the true modifier.
See also	The `LEADER_DEFAULTS` element introduced by xmldefault could be used for this purpose. However, currently that element is deleted once XML loading has finished.

AdvCiv	*BBAI/ Civ Accelerator mod*
After computing an attitude breakdown, the cached total relations value is compared with the sum of the breakdown. If the two values aren't equal, an assertion message is displayed, the attitude cache is updated and the breakdown is recomputed before being displayed. None of the above is done in networked games because it would result in an asynchronous cache update.	The total relations value shown above an attitude breakdown is read from the attitude cache, whereas the modifiers in the breakdown are computed on the fly. If the cache is out of date (which should ideally never happen), this becomes apparent when the displayed total relations value doesn't match the sum of the modifiers. Actually, only with the SHA-Mod the inconsistency is apparent – because, otherwise, hidden modifiers could be responsible for the discrepancy.
Rationale	Consistency problems with the attitude cache keep cropping up now and then, and any changes to the computation of the relations modifiers can introduce new errors of this kind. The above hides these errors from the end user and makes them easier to spot and resolve when testing with an attached debugger.
Tbd.	Check if I can make use of K-Mod's `CvPlayerAI::AI_changeCachedAttitude` to avoid recomputing all relations modifiers all the time.
004r	Report resources discovered on unclaimed tiles

AdvCiv	*BtS*
When a tech reveals a resource, sources in unowned and Barbarian territory are reported as well, and there's a special message when no source was discovered or only on tiles owned by rival teams. Resources on the territory of the player's vassals are also reported.	Only resources on the player's current territory are reported.
Rationale	All sources could be interesting to the player, including those owned by other teams, but mustn't flood the screen with messages. The warning about no source probably matters most for Animal Husbandry; easy to forget to check whether Horses have been revealed anywhere.
004s	Economy ("GNP") curve doesn't count culture and espionage; yield curves show moving average
See also	091: Changes to score graph

AdvCiv	K-Mod
The Economy graph on the Info screen is named "GNP (Gold+Research)" and shows the net sum of the gold generated by all cities plus the sum of the research generated by all cities. This sum doesn't include any costs (e.g. city maintenance or civic upkeep). It does include (no change) all gold and research modifiers (e.g. Bank, Library) and commerce from trade routes, buildings (Shrine) and specialists.	Called "GNP (Gold)" and shows the sum of all commerce produced by cities, including culture and espionage. (So far, as in BtS.) Gold is reduced by the sum of all gold costs, including inflation. Research is increased based on modifiers that apply to the currently researched tech: from tech diffusion and from knowing multiple prerequisite techs.
All the yield curves show moving averages based on three samples.	All curves show the game state at a given turn, not an average.
Economy and Industry ("goods produced") curves show a (very short) straight line for revolution turns.	When a revolution turn is sampled for one of the curves, a sharp, brief decline is shown.

Rationale	Culture and Espionage have their own curves. Both can reach far higher rates than research and gold normally do; e.g. a settled Great Spy produces 12 espionage per turn, a settled Great Scientist only 6 research. This makes it hard to tell from the Economy curve if a civ has a strong economy or if it's merely focusing on culture or espionage. A single curve for gold and research is OK; they're on the same scale.
	By including costs, the curve would essentially become a research curve, which is probably more useful than some commerce curve, but then it would seem like a "Gold" curve is missing. Excluding costs is also consistent with the Agriculture curve ("crop yield"), which doesn't include food consumption.
	As for research modifiers: Any modifiers that apply only to the currently researched tech (which can change any turn) are misleading.
	Moving average: The Economy and Industry curve fluctuate a lot in BtS, to the point of being hard to read, and removing culture and espionage (which rarely fluctuate) from the Economy curve makes it worse. (One cause of the fluctuation is AI juggling of citizens.) Apart from usability, the average is also supposed to obscure the changes from one turn to the next a bit. I can't make much sense of them, but it might be possible to derive some specific information, e.g. about AI war planning or wonder building, that the player isn't supposed to have. The power curve has that issue too; however, it might be confusing if the enemy power curve doesn't immediately drop after defeating an enemy stack.
	Revolution turns: The negative spikes are ugly, and confusing because revolution turns aren't always sampled when drawing the curves.
Tbd.	"GNP" is a bad name because "Goods produced" would have to be included in a GNP. Showing yield icons instead of the words "production", "culture" etc. would be nice, but it seems that the Python drop-down menus don't support icons in their labels.
See also	077 makes lets the "Demographics" tab use the moving averages of the "Graphs" tab. That change also uses some of the same game text as 004s.
	038 uses the GNP curve to determine which civs are "wealthy".
	132 uses the GNP curve to predict the cost of anarchy.
	UWAI uses the GNP curve for converting war utility into trade value and vice versa, and the production curve for estimating military build-up.

Moved the player statistics into a separate class and switched to a more efficient data structure.\n\nAs a side-effect, the graphs on the Info tab no	The mapping of turn numbers to statistical data is handled by an STL hash map.

longer show a discontinuity when the active player is switched through Alt+Z.		
Rationale	Now that the AI accesses this data, the access needs to be fast.	
See also	advc.opt: Other performance optimizations.	

004t	Cick-to-exit-city-screen disabled	
AdvCiv		*BtS*
If the "Click Map to Exit" option is disabled (City tab of the BUG menu, enabled by default), then the city screen can't be (accidentally or deliberately) exited through a left-click on a tile outside the city radius. *Double* clicking outside the city radius always exits, and all other methods for exiting the city screen also work as before. `CITY_SCREEN_CLICK_WILL_EXIT` in XML no longer has any effect. The flashing text "Press <Esc> to return" is no longer shown on the city screen.		The city screen can always be exited through middle-click, left-click on the mini-map, Esc, Enter, Space, by selecting a garrisoned unit or by accessing an Advisor screen through a function key. Left-click on a non-workable tile also exits the screen; since the BtS expansion, there is an XML switch to disable this.
Rationale	Happens too easily by accident. Especially annoying when the city has just been given production orders because, then, the next popup will appear before the player gets a chance to re-enter the proper city screen. Apparently, the original developers wanted the game to be playable without a keyboard and with just one or two mouse buttons, and I agree with that – fortunately, one can still exit the city screen by clicking on the mini-map. Not very intuitive, but neither is clicking on an unworkable tile. Since AdvCiv 1.0, the option is enabled by default (i.e. for new installations) because players kept getting confused by not being abloe to exit and weren't aware of the option.	
Config	This change has (understandably) kept bothering players, so I've moved the option from XML to the BUG menu (City tab).	
See also	One of the CFC posts inquiring about the click-to-exit behavior. I think there were two other such posts over the years, one by keldath. 002n disables the flashing end-turn message.	
Double click on the city tile no longer exits the city screen. Instead, right-click on the city tile (city center) exits the city screen. When right-click is used for exiting, the camera centers on the city tile afterwards.		When the city screen is up, a single click on the city tile re-enables citizen automation if it has been disabled. A double click does that too before exiting the city screen. If a working city has been assigned to a tile manually, right-click undoes this assignment. After exiting the city screen, the camera always center on the selected units (if any).
Rationale	Normally, a player that isn't aware of the single click behavior will notice that citizens are being reassigned, but if the city screen is exited in addition, the player may easily not notice his/her accident.	

	Right-click on the city center didn't do anything previously because the working city of the city tile itself can't be changed. The camera jump is usually annoying, however, when inspecting the capital through the flag button (i.e. without moving the camera over the capital first), then jumping back to the selected units is welcome. So I don't want to disable that. (Ideally, perhaps, the camera would always return to its previous locations, but I don't think that can be implemented.)
Tbd.	Can I somehow tell when a city screen has been entered from the Domestic Advisor? In that case, exit via right-click should work the same way as the other methods, i.e. should not move the camera.

004u	More info in message about Great General (GG) death
AdvCiv	*BtS*
The message about the death of a GG states the owner of the GG and the owner of the unit that killed the GG, and indicates the location of the battle between the two on the map (if it is revealed).	The player only learns the name of the GG. Can only search the Event Log for the birth message (if it hasn't expired) to find out the GG's owner.
Show the message only to players who have met the owner of the Great General.	Shown to all players.
Rationale	Debatable if the location should be indicated; more flavorful this way.
The regular message about a unit killed in an attack doesn't mentioned an attached Great Warlord by name, i.e. might just say "Your Cavalry has died attacking a Rifleman." Only the GG death message mentions the name of the Warlord.	Both messages mention the Warlord by name, and the regular message says e.g. "Your Heinz Guderian has died attacking a Rifleman."
Rationale	The BtS messages are more overtly redundant and not grammatically correct. The grammar is still incorrect in messages about a Warlord surviving combat. These messages are quite frequent, so I've tried putting the "Your %s1_UnitName" phrase into a separate key, but this would've caused problems with gender agreement in the Romance languages.

004v	Changes to scoreboard text
See also	[007](#) makes some changes to the text shown in Debug mode. [106d](#) changes BUG settings concerning the scoreboard. [190d](#) avoids giving away the civ and leader of unmet human rivals
AdvCiv	*BtS*
The hint about Ctrl for trade table and Alt for DoW are merged into a single line.	Two lines of UI help, one above the line about worst enemy, the other below.
Rationale	Cleaner this way.
In multiplayer, AI civs are no longer marked with "[AI]" on the scoreboard.	
Rationale	Square brackets are for highlighting the active player. And it should be clear enough from the leader names which ones are AI-controlled.
See also	[155](#) color codes team membership on the scoreboard.
AdvCiv	*BUG*
When the option to show defeated civs on the	Seems to work only for teams that were met

scoreboard is enabled, then even players that hadn't met a dead team see it on the scoreboard. (The option remains disabled by default.)	before being defeated.

Rationale	Defeated civs are announced by name, so this change doesn't "spoil" anything.
Credits	Based on these two Kek-Mod commits (but implemented differently). (Not merged: "*Player names of unknown players are also concealed during advanced start.*" I don't really understand the code or the problem it solves.)

004w	Removal of unnecessary help text
AdvCiv	*BtS*
For buildings already completed, don't show requirements in help text, and don't show if it's a unique building, if it founds a corporation, if it starts a Golden Age, if it centers the map, if it's required for another building, if it can be built by a Great Person.	Help text for buildings in a city's building list shows e.g. that the Palace requires at least 4 cities and that the Trading Post is a unique building replacing Lighthouse.
Help text for wonders and projects only says how many instances are allowed if more than 1 instance is allowed, which is only the case for some spaceship parts.	Says e.g. "Wold Wonder: 1 Allowed" or "(National Wonder: 0 Left)".
The remaining slots for National Wonders are only shown in help text on the city production buttons, not on the list of completed buildings, and e.g. "2 Left" means that two more national wonders can be built in the selected city.	"2 Left" would mean that two more cities can build the given wonder. The number of remaining national wonders per city is only shown once it is maxed out.
Show "0 left" in red when a wonder is already in production in another city.	The reason why a building can't be constructed is normally shown in red but not in this case.
Civiliopedia no longer shows whether a building appears for free when starting in a later era.	
Some abilities that require a tech are not shown in help text when the player's era is more than 1 behind the tech era.	E.g. Work Boat: "Can Create Oil Rig" shown in the Ancient era.
Active dual deals (peace treaty, OB, DP) shown as e.g. "Open Borders with Alexander".	"Open Borders to Alexander for Open Borders"
Rationale	This text is at best distracting, at worst misleading. The number of remaining national wonders per city is especially helpful for One-City Challenge.
See also	004a removes "can be researched by a" from technologies. 002b (larger fonts) makes it a bit more urgent to reduce the amount of help text. History Rewritten explicitly displays the remaining slots for national wonders on the city screen (screenshot). Shouldn't be necessary in AdvCiv (now that the help text is improved).
Help text for obsolete buildings in the building list of a city says that it's obsolete. The word "obsolete" is only shown in red when a building is already obsolete. Otherwise e.g.	No indication in the building list whether a building is obsolete.

"Obsolete with Scientific Method" Tech requirement of Temple is shown in hover text.	"Obsolete with Scientific Method" It's a special building requirement; must've been overlooked somehow.

Rationale	While I was at it.

Help text for resources overhauled – some inappropriate text removed but also a fair amount of text added.

See also	In part dependent on 073, which merges the trade denial hovers from BULL and replaces one of the columns of the "Resources" tab on the Foreign Advisor screen.
	036 also adds information to the resource hover text (only on the Foreign Advisor).
	003p caches the help text for the Resource layer. Without that change, there would be a noticeable delay whenever the Resource layer is enabled or updated. 003b also improves performance a bit by caching the number of teams ever alive. (That number is needed by `CvGame::canConstruct`.)
	047 revises help text for resource tiles.

Help text for resources omits health and happiness effects from obsolete buildings, from buildings whose tech era is more than one era ahead of the player's current era and from buildings that require religions which the player doesn't have access to.	E.g. shows "+1 health from Supermarket" for Pig already at the start of the game, and lists all seven Cathedrals for Incense.
The tech to reveal a resource is only shown if the player doesn't have that tech yet.	Will say "Revealed by Animal Husbandry" for Horse for the entire game.
Tile yields of resources are only shown in Civilopedia text and when inspecting resources on the map.	Always shown, but K-Mod has added "on plot" to clarify that the yield can't be traded.
On the main map and city screen, the number of resources owned by the active player is stated as e.g. "(we have 2)", and, on the city screen, it's only shown when the number isn't 1.	"(firpo has 1)"
On the "Resources" tab, the help text for the export columns (cf. 073) says how many resources the other players own.	The other player's resource counts are only shown on the Trade screen.

See also	074 excludes must-be-joking resources from the Trade screen, meaning that the AI resource counts aren't necessarily visible there.

Tbd.	For religions, it would be better to list the religious buildings under a single name, e.g. "+1 happiness from Cathedral" with all religion icons listed in parentheses after "Cathedral" (all in one line). Eventually, I want to move the happiness ability from the Cathedral buildings to the Temple buildings, and then it would be better to show it from the beginning of the game, even if no religion has been founded/ spread yet. For a clean implementation of this, the happiness ability should be moved to `CIV4SpecialBuildingInfos.xml`.
	When trading (Trade screen or "Resources" tab), the surplus health and happiness of the capital should perhaps be shown in help text. E.g. "Sheep +1 health (Rome has +5 health)." I think BULL does something similar but uses the smallest surplus among the active player's cities.

Help text for resources shows buildings, projects and units that are affected by strategic resources,	Only tile yield, tech-to-reveal, active corporations, happiness and health effects are shown in help

so long as they're not too far in the future, obsolete or in some other way irrelevant. The names of boosted buildings that are currently under construction are highlighted; exception: Not in the resource "bubble" text on the main interface because it's too difficult to keep that up-to-date (cf. 003p).	text.
Rationale	After removing tile yield and tech-to-reveal info in most situations, the strategic resources had only their names left. The new info should be helpful when deciding whether to trade away a non-surplus strategic resource.

When the city screen is up, the effects shown in resource help text are based on which buildings and units the selected city can produce. Buildings that the city already has are shown in green, and buildings that the city doesn't have but could have someday, in red, e.g. "Wine +1 happiness; +1 health from Grocer".	Help text for resources is not based on the selected city. Will e.g. say "Clam +1 health; +1 health with Harbor" in a landlocked city that doesn't have a Harbor and won't ever have one.

Shortened help text for production modifiers from resources and traits by replacing the word "production" with the hammer icon and omitting the word "speed".

Production modifiers from traits are shown in hover text for building and unit buttons (e.g. on the city screen) if they apply, i.e. if the active player has the proper trait.	All production modifiers from traits are shown in Civilopedia (regardless of the active player's traits), but, on the city screen, only production modifiers from resources are shown. The impact of traits can only be inferred from the predicted production turns.
Rationale	The trait production boosts are difficult to remember, so highlighting them on the city screen is very helpful I think.

Stats on unit pane:

Don't show a row for movement points for ICBM; for air units, replace that row with a row for air range.	Immobile units (ICBM) and air units are show as having 1 movement point. Air range isn't shown.
Don't show a row for XP level when the selected unit has 0 XP.	A level is always shown; 1 by default.
See also	004y (unit stats on Sevopedia) about the movement / range info.
Rationale	Level 1 generally isn't interesting; even a bit confusing when a player isn't sure if levels start at 0 (they don't). Still, once a unit has XP toward level 2, it's probably better not to hide the level. Moreover, the XP check is a good way to always hide the level of missiles and civilian units. (Well, not absolutely always; if a mod-mod allows e.g. Spies to gaher XP, then the level info will appear.)

004x	Cancel redundant minimized popups; other changes to civics/religion popups

No minimized popups are canceled in networked multiplayer games.

Rationale	I can't work out the proper timing. If the multiplayer checks added in this Git commit are removed, then popups will sometimes appear multiple times. Simply adding a delay to the cancellation (using the `setUpdateTimer` function added for 004j and a queue of popups to be canceled) doesn't solve the problem either; too many popups get killed then.
AdvCiv	*BtS*

When a choose-tech popup is launched or the player chooses a tech, all pending choose-tech popups are killed. When a change-civics popup is launched or civics are changed, all pending change-civics popups are killed. When a change-religion popup is launched or the player converts to a religion, all pending change-religion popups are killed. When anarchy breaks out, all choose-production and choose-tech popups are killed. They reappear when order is restored.	When research finishes on a tech and no tech is queued for research, a choose-tech popup is launched and appears minimized at the start of the next turn (if playing with the "minimize popups" option; otherwise there isn't a problem). For each additional tech that the player receives through e.g. tech trade before choosing his/her next tech, another minimized popup is launched. All but the latest popup will show outdated research options (not taking into account all the additional techs discovered). Once the next tech is chosen, any remaining popups disappear once maximized. Similarly, multiple change-civics and change-religion popups can appear and don't disappear when the player changes civics/ religion through an advisor screen. I don't think multiple choose-production popups for the same city can appear, but the single popup fails to disappear when the player chooses production through the city screen.
Rationale	I'd actually prefer to allow multiple change-civics and change-religion popups, but it's difficult to do this consistently as there appears to be a mechanism in the EXE for preventing multiple change-civics popups, it just doesn't always work. E.g. after discovering Feudalism, BtS shows only one popup for Serfdom and none for Vassalage (intended behavior I think, but I'd like it better to also have a popup for Vassalage), but when trading for, say, Code of Laws while the change-to-Serfdom popup is waiting, another change-civics popup will appear for Caste System (OK, but inconsistent with the missing Vassalage popup).
See also	
During anarchy and before the first city is founded, civs have 0 research rate and production rate, and the research bar shows the remaining anarchy turns even if no tech is currently selected. All game text that normally shows the remaining research or production turns doesn't show that information when the rate is 0. When the `BASE_RESEARCH_RATE` is active (i.e. once a city has been founded and while not in anarchy), the research rate shown on the main interface includes the 1 free beaker.	The `BASE_RESEARCH_RATE` (set to 1 in the Vanilla `GlobalDefines.xml`) applies even during anarchy. When no tech is chosen during anarchy, choose-tech buttons are shown on the research bar. Game text shows the remaining research and production turns based on a rate of 1 per turn during anarchy. The research rate on the main interface shows only research from city commerce.
Rationale	During anarchy, the turns to complete can't be properly computed, therefore, research and city production shouldn't be chosen. Setting the research rate to 0 makes it easier to communicate this, and makes sure that there is no incentive to choose research.
Tbd.	Get rid of the `BASE_RESEARCH_RATE`. Can instead increase the commerce from Palace by 1 or even 2, which will make Gold, Gems and Silver less powerful. That said, commerce modifiers in the capital would become even more powerful this way, so perhaps give Palace 10 commerce minus 1 for every other city, or 5 + max{0, 5 - number of other cities}.

When the Pyramids or Shwedagon Paya is completed, a change-civics popup is shown unless the player already had access to all Government/ Religion civics.	Buildings that unlock an entire column of civics don't trigger a change-civics popup.
When a new civic becomes available (through a technology or a wonder) at a time when no revolution is possible, then the popup is delayed until the player can change civics again.	
When the game starts in an era later than Ancient, a change-civics popup is shown on the initial turn.	No such popup. It's normally best to change civics on the initial turn – but easy to forget.

Rationale	I've forgotten about switching civics after completing the Pyramids one time too many.
Tbd.	Should also delay change-religion popups. Could then, as part of an overhaul of the leader traits, restrict the fiddly ability of the Spiritual trait so that a free revolution is allowed only on a turn on which a change-civics or change-religion popup occurs (and perhaps once per era in addition).

When the active player enters the Civics screen, all minimized change-civics popups of that player are canceled. Same for the Religion screen and change-religion popups.

Rationale	The player probably noticed the new civic or religion when or before viewing the respective screen, making the popup reminder unnecessary.

004y	Changes to Civilopedia formatting; Sevopedia
See also	905b turns the "Allows" box in articles about resources into a "Units" box in order to accommodate the extra moves that ships can receive from resources.
Tbd.	Mongoose Mod may include some improvements for the link and jump code that I could merge. See the mod's changelog and v4.1 release notes.
	Perhaps information shown in hover text shouldn't take into account the current game state when the Civilopedia (Sevopedia or original) is opened within an ongoing game. For example, hover text for religious buildings currently shows a gold rate when the active (human) player owns the Spiral Minaret. CvGameTextMgr in the DLL can tell whether help text is being composed for a Civilopedia article and whether Civilopedia is being accessed from the main menu, but it doesn't the context of hover text. That could be remedied by passing a special value as `iData2` (currently unused) to the help text widgets in the 'Pedia Python code. CvGameTextMgr could then treat 'Pedia hover text the same in an ongoing game as when accessing the 'Pedia from the main menu (by treating the active player as `NO_PLAYER`). All that being said, perhaps it's better to reflect the current game state in 'Pedia help text as is the status quo; the static abilities are still shown in the actual articles.

AdvCiv	*Sevopedia*
Increased the width and height of the Sevopedia screen on resolutions wider than 1024 pixels.	The code for the dimensions is mostly copied from the original Civilopedia. The size of the main panel doesn't scale with the screen dimensions.

See also	CFC post on the current status. And two more. For what it's worth, the More Naval AI mod has also enlarged Sevopedia.
Config	`bWideScreen` and `bFullScreen` flags in `SevoPediaMain.py`
Items that have XML data only for technical	Hills appear under "Terrain" with the only

reasons (`GraphicalOnly`) are not shown in the Sevopedia; e.g. Hills are not listed under "Terrain". (Perhaps this is the only example.)	information being "can't found cities".
Tbd.	I wouldn't mind listing Hills (and Peak) under "Terrain" – currently they're not listed anywhere –, but showing sensible information would take some work.
In articles about technologies, the "Civilizations" box isn't shown if it would be empty.	The "Civilizations" box lists the civs that start with the respective tech. It's shown for all techs and usually empty.
Rationale	There are other empty boxes in Civilopedia too, but this one doesn't have a self-explanatory name, so it can be confusing if empty.
On terrain improvement pages: "Improvements" box renamed to "Improvement Yields" and "Bonus Yields" to "Bonus Resources".	The first box lists base yields and yield enabled by technologies, the second one lists special yields from bonus resources.
Rationale	The first box needs to say something with "yields" for clarity and then the second one shouldn't also say "yields".

Civilopedia lists (when clicking on a top-level category like "Units" or "Wonders")

AdvCiv	*BtS*
Show production costs of units in hover text (but not under "Special Abilities"). When Civilopedia (or Sevopedia) is accessed from the main menu, show number of required buildings as e.g. "Requires University: 4-6". Also show other numbers that depend on the map size as a range.	Only shown inside the unit articles. The number from `Civ4BuildingInfos.xml` is shown, which corresponds to Duel map size. E.g. "Requires University (4 Total)".
See also	[140](#): Changes to map size adjustment multipliers. [008](#): Changes to Civilopedia content

AdvCiv	*K-Mod*
Sevopedia (with alphabetical sorting) enabled by default. When entering Civilopedia from the main menu without having started or loaded a game first, Sevopedia is shown regardless of the setting in the BUG options menu. Sevopedia then remains enabled until Civ 4 is restarted. Added a warning about this to the hover text of the Sevopedia option on the BUG menu. Sevopedia doesn't list e.g. Ruins under "Improvements"; only improvements that civs can actually build.	Disabled by default. If Sevopedia is enabled from the BUG options menu, the BtS Civilopedia is still shown when entering Civilopedia directly from the main menu. The Sevopedia setting in the BUG options is then permanently ignored: Sevopedia remains disabled even after restarting Civ 4; only toggling the checkmark in the BUG options menu twice brings it back.
Credits	Copied [this](#) bugfix by EmperorFool. I also had to copy his [makeOptionId](#) function. (The bugfix was later reverted, perhaps because it doesn't fully solve the problem.)
See also	More or less fixes [this](#) open K-Mod issue.
Tbd.	Would be nice to use the `INTERFACE_GENERAL_CIVILOPEDIA_ICON` instead of the commerce icon for the Hints and Concepts sections on the index. That would have to happen in `SevoPediaIndex.py` under `type == "Concept"`, but can I simply use `addDDSGFC` there?

Unit stats: Don't show strength for nukes; don't show movement points for air units; show "immobile" for ICBM and "unlimited" range.	Nukes are shown as having 0 strength; air units and immobile units (ICBM) are shown as having 1 movement point; ICBM is shown as having 0 range.
Rationale	The Sevopedia info was, in a way, technically correct – but confusing. Showing meaningful ICBM stats could actually be useful for new/ returning players; not obvious how that unit works.
See also	004w makes similar changes on the main interface. rheinig's mod (see advc.rh) shows "immobile" as a special unit ability instead.
Show production cost as the final stat.	AIr range is shown below production cost.
Rationale	Cost shouldn't be mixed in with the benefits.
Put a gray panel behind the list of hints; same style as in the BtS Civilopedia.	
Rationale	The white text was too hard to read on the beige background.
The box with the strategy help and historical background has the title "Background" and the historical background section has the heading "History".	The box has the title "History" and the historical background section has the heading "Background".
Rationale	Strategy advice doesn't fit under "History".
Config	Through `Civ4GameText_advc.xml`
Sevopedia categories for keyboard shortcuts and leader traits restored. Removed mention of shortcuts that AdvCiv doesn't support. Removed the trait button icons. Still disabled: Strategy advice for the traits and the category for strategy articles.	K-Mod has disabled these; explanation. There is a GameFont icon for each trait shown in the list of traits (I didn't remove those icons) and a button icon shown in the article. All the icons are reused, e.g. the "Heal" action button for the Expansive trait.
Rationale	The traits category comes with strategy advice, which is tedious to keep updated, so I've kept that aspect disabled. Trait buttons with hover text could be useful elsewhere in the UI, but unique icons would be needed for that, and currently there is no hover text, so the Sevopedia trait buttons serve no real purpose.
Config	To remove these categories again, it's probably easiest to comment out one or two lines in `setPediaCommonWidgets` (`SevoPediaMain.py`). Adding the `TraitsPedia_CIV4GameText.xml` file from BUG should suffice to restore the strategy advice on traits.
Credits	The shortcut lists are originally Ekmek's work, the traits Sisiutil's. A couple of shortcuts that I've added I got from dj_anion's BtS reference guide.
See also	002b: Had to add/remove some tabs to get the shortcut info properly aligned.
Tbd.	I've only made stylistic changes to the English version. The other languages probably include some mistakes (because their keyboard layouts are different). The shortcuts should be ordered by effect, not by the key combination. People don't consult Civilopedia to check what a particular key combination does, they want to know whether there's a hotkey for a certain effect. Would be nice to add a subcategory for cheat/ debug shortcuts (see 007b).
Added a box named "Changes" for pointing out balance changes to traits made by K-Mod and	

AdvCiv. And articles about civics now include strategy help texts (previously only shown as a short summary in change-civic popups) with notes about K-Mod and AdvCiv changes appended.	
Tbd.	Several other categories of articles still lack a space for balance changes: promotions, improvements, specialists, projects, corporations, leaders (would be nice to mention significant changes to the AI behavior of individual leaders).
Opening an Advisor screen no longer causes the Civilopedia button to be hidden.	
Rationale	Hiding the button means that an extra click is required to get from an Advisor to the Civilopedia. Ideally, the Advisor screens should include direct Civilopedia links to all relevant information, but that isn't currently the case.

004z	Changes to layers	
See also	[009c](#) (removal of Map Finder) makes room for the options on the BUG menu. [004m](#) stores the currently active layer in savegames.	
AdvCiv		*BtS*
The Resource layer always shows all resources; no option in Globe view for filtering the shown resources.		In Globe view, all resources are shown by default, and a selection box opens for showing only happiness, only health or only strategic resources.
The Unit layer shows all units by default regardless of Globe view.		The unit layer shows "Enemies in Territory" by default. In Globe view "All Units" is the default, and several other options can be selected (still the case in AdvCiv). It appears that the "Enemies in Territory" option was added by the BtS expansion.
Config	Option on the "Map" tab of the BUG menu. The default option for the Unit layer can only be changed through the DLL (a hack was required). Can hide the Unit layer options through `GlobalDefines_advc.xml`.	
Rationale	The Resource options seem almost useless to me. Could make it easier to check which rivals have access to a particular resource, but I've never once used it for that. I'm less sure about the Unit options. Even when one is only interested in e.g. enemy units, showing allied units (in a different color) along with enemies doesn't seem like much of a distraction; after all, they can't occupy the same tiles. That said, showing only "Domestic" units is useful for locating Spies and Workers, though I think few players ever use this. In any case, "Enemies in Territory" isn't a good default option in Standard view as, normally, there are none. Makes it seem like the Unit layer is broken (help text doesn't say which units are shown).	
Tbd.	It would be nice if the Unit layer could somehow communicate stack sizes more clearly and not just in Globe view.	
When a layer without options is selected, the scoreboard is shown in Globe view.		The scoreboard is never shown in Globe view.
Config	Option on the "Map" tab of the BUG menu	
Rationale	Should be possible to access to the scoreboard in Globe view, e.g. for following [all-AI games](#) from a high viewpoint. When there are no options to display, the space might as well be used for the scoreboard.	
Tbd.	Would be nice to have a button that toggles the scoreboard in Globe view. Currently, one has to go back to Standard view in order to disable the scoreboard. Above the giant flag button, there's still room for plenty of layer buttons. Doesn't seem trivial to	

implement though.	
The Resource layer also highlights goody huts when the "All Resources" option is selected. The unit action recommendations (UAR) option causes huts to be highlighted when any nearby combat unit is selected (can be a Recon unit, but doesn't have to be).	Highlights only resources. Tribal Villages aren't highlighted by any layer. If UAR are enabled, Tribal Villages are highlighted with a blue circle when a nearby Recon unit is selected.

Rationale/ See also	Normal map scripts don't place a hut and a resource in the same tile, so the Resource layer seems like a good place for highlighting huts. [315](#) increases the size of the goody hut 3D model, but they're still a bit hard to see on Forest tiles.
	Until AdvCiv 0.99, the Resource layer had shown huts only when UAR was disabled. A circle *and* an indicator bubble is arguably too much highlighting, but let's let the player decide that.
	In any case, entering a nearby hut with a Warrior is usually a good idea, so UAR shouldn't be restricted to Scouts.
Config	Option on the "Map" tab of the BUG menu

The "normalization" step after the assignment of starting locations tries to avoid placing resources on goody huts and removes the hut if it can't be avoided.	The only way that a resource can coexist with a goody hut on a map generated by one of the official or bundled map scripts is through the normalization step.

Rationale	Don't want resources and huts in the same place because the resource layer can't show both.
See also	[108](#) deals with other changes to the normalization step.

When in Globe view, layer icons ("indicators", "bubbles") are shown at a smaller diameter than in Standard (non-Globe) view.	Resource icons are the same size in Globe view as in Standard view. It might be that their size relative to the size of a tile increases with the screen resolution; one CFC user describes the icons as "ridiculously big at 1920x1200". ([link](#))

Rationale	Unfortunately, there appears to be [no way](#) to shrink the indicators in Standard view. Shrinking them in Globe view is better than nothing.
	Based on asaf's [Fourth Yield](#) mod, it might – with a lot of work – be possible to show the resource icons as overlayed symbols like the Yield Display, but I've made a mockup and it looked uglier and less functional than the large bubbles. One remote possibility that I haven't really looked into: Using `CvDLLFlagEntityIFaceBase` to place smaller resource icons on flag poles.
See also	merijn_v1 created [this](#) mod comp to allow yields greater than 9 to be displayed on the map. That's not going to help with the resource display, but it's essentially a bugfix that I might want to adopt. Through random events, yields of 10 production or commerce are possible. However, I don't care much for random events and yield display, and the mod would add nearly 100 KB to my download size. If I do merge it, it would be easiest from [this](#) Git commit (code looks a bit redundant though?).
Tbd.	Something needs to be done about the enormous indicator bubbles on large monitors. In principle, it's possible to inject code into the EXE – but one has to pinpoint where (disassembly) the EXE sets the indicator bubbles, and expect that this location is pretty far removed from the DLL in the call graph. billw2015 might be able to do it, but I'm not.
	A different approach would be to embed resource icons as flat billboards in the resource (3D) graphics. Players could then still switch the indicators on and off by

swapping the modified graphics and the original graphics, but the indicators couldn't move based on the camera position. Making a test with a single resource shouldn't be so difficult, but I've no experience with modifying the `.kfm` files.	
The Trade layer colors all revealed tiles according to the trade network grouping except some very small groups like water tiles enclosed by shelf ice.	The Trade layer colors only tiles owned by the (human) player's team.
Cities that are actively visible to the player (i.e. not fogged) and not connected to their owner's capital are marked in black.	No special marker for disconnected cities.

Rationale	I think this layer is used (if ever) for getting a better understanding of the trade group rules. Showing trade along unowned coasts (with Sailing) should be an improvement. It would be nice if the layer could also show the trade networks of rivals – in BtS, the coloring is entirely based on the player's civ. The problem with showing rival networks is that it can give away information about unrevealed coastlines and routes. Therefore, I'm only providing info about currently visible cities. The black mark should be helpful for verifying that an enemy city has been cut off from the capital, e.g. through a naval blockade.
See also	124 allows only revealed routes to carry trade. Without this, the Trade layer coloring of tiles owned by other civs would give away routes on fogged tiles.
Config	`GlobalDefines_advc.xml`

The Unit layer option for showing non-military units is called "Civilians". The indicator bubbles show a non-military unit. If multiple non-military units are present, the shown unit is selected based on a priority list (from highest to lowest): Great Person, Settler, Spy, Missionary/Executive, Worker/ Work Boat.	Called "Domestics". Some commented-out code suggests that, in some earlier version (Warlords?), only units within the active player's borders were shown, but BtS highlights all tiles with visible non-military units. The plot indicator on a highlighted tile shows the tile's top defender. That's a military unit when the tiles contains both military and non-military units. Air units aren't recognized as military units.
Air units are considered military units.	

Rationale	The name "Domestics" doesn't fit anymore in BtS, and it shouldn't show bubbles with military units – it's counterintuitive and makes it difficult to locate Spies (which are usually in cities together with military units).

For fogged tiles, the culture layer uses only the color of the last known owner at a fixed brightness (alpha value).	The culture layer doesn't distinguish between fogged and actively visible tiles. Shows the colors of up to four civs that have culture in the tile. The brightness is set based on the culture values. Outside the culture layer, culture percentages are shown in hover text for actively visible tiles, whereas, for fogged tiles, only the last known owner is indicated (through cultural borders).

Rationale	There's no need for letting the culture layer (partially) bypass the fog-of-war rules. In fact, the map looks tidier with mono-colored tiles in the fog of war.
	Perhaps a significant culture percentage (say, 20%) should grant visibility. Then again, it's not too unrealistic that the ruler of a civilization wouldn't know all the places that the civilization's culture (or ethnicity) has spread to.
See also	001f reveals the new owner of a fogged city after conquest by a third party. In that case, BtS is quite restrictive about revealing cultural ownership in fogged tiles.
Tbd.	Add indicator bubbles for (actively visible) cities with a positive revolt chance (icon

	civics_popup.dds?), cities in occupation (that orange fist icon), with hover text stating the revolt probability and occupation countdown, and cities relevant for culture victory (how to measure that?), especially foreign cities, which aren't covered by the Domestic Advisor. Would have to review the code that sets the globe layers dirty-bit (see 003p).

AdvCiv	K-Mod
The culture layer shows the tile owner's color even if that civ has less than 20% tile culture, e.g. right after conquering a faraway city.	Only the colors of civs with at least 20% tile culture are shown.
Rationale	Since borders aren't shown in Globe view (and can't be shown through the SDK), the Culture layer is the only way to highlight territories in Globe view. Too easy to overlook a recently conquered area when it is shown entirely in other civs' colors.
See also	099f shows culture on unowned tiles in tile hover.

AdvCiv	BtS
When the culture of multiple civs is displayed on a tile (see conditions above), the colored area of each civ corresponds to that civ's culture percentage. For example, a tile that is 75% Egyptian and 25% Japanese is shown as 3 parts yellow and 1 part red.	The colored area is split evenly among the civs, with ties broken in favor of the lower civ id. The percentages are only indicated through brightness.

2 parts yellow, 2 parts red |
| *Rationale* | The differences in brightness are very subtle, almost useless really for multi-colored tiles. After my change, it's easy to tell which tiles are contested, and which ones have only a significant minority culture. |
| *Credits* | Prompted by this old CFC succession game. Several players complained about the culture layer in K-Mod. |

005	Minor flavor changes
"Sumeria" changed to "Sumer"	
Rationale	Seems to be much more common (also in German and Italian). Wiktionary lists "Sumeria" only as "(archaic, poetic) Sumer." And brevity is always good.
"Pacal II" renamed to "Pacal"	
Rationale	According to Wikipedia, the earlier Pacal wasn't necessarily a king; the article refers to our Pacal (the Great) simply as "Pacal." Simpler names are better. I haven't renamed "Montezuma" to "Montezuma II" – although he is as much a second as Ramesses II, Mehmed II, Suryavarman II and Joao II – because it would be a complication and also because the the proper name would really be "Moctezuma II," which might confuse players.
"Asoka" renamed to "Ashoka"	
See also	Discussion (near the middle of the post)

(I wasn't going to make this change at first, but it doesn't seem too difficult after all too get used to the spelling change.) |
| "Justinian I" renamed to "Justinian" | |
| *Rationale* | Justinian II is also an important figure, but not as important as Justinian the Great, and also isn't the kind of ruler that would be included in a Civ game. So players should have no difficulty deciding which Justinian is meant, and the name looks a bit too long |

	with the regnal number.
	About Cyrus II, it's more difficult to decide. Leaving out a regnal number other than "I" seems problematic – though there is precedent in Montezuma, and Cyrus I is much less significant than Cyrus the Great.
005a	Leader personality tweaks

AdvCiv	*BtS*
`MaxWarMinAdjacentLandPercent=2` for Bismarck. While he eventually gave up his reluctance to off-shore colonies, he was never keen on distant parts of Europe (let alone Asia): "The Balkans are not worth the healthy bones of a single Pomeranian grenadier." The Realism Invictus mod even sets it to 4.	Was 0. It's a preference for long-distance land wars, ranging between 0 and 4. 0 means Bismarck treats neighboring civs no different from other civs on the same continent when it comes to war planning.
Also lowered his attitude threshold for defensive pacts to Cautious (realpolitik, Triple Alliance), and increased his weight for diplomatic victory and lowered the one for domination victory. Conversely, gave Brennus higher weight for domination and lowered weight for diplo.	Threshold is at Pleased, like most other leaders. Victory condition weights have been added by BBAI. I don't think they actually have a big impact on the game.
While I was at it (further minor changes): Shifted Catherine's weights from conquest toward domination and science. Added a bit of science weight to Ramesses, and reduced his conquest weight. Increased Saladin's weight for diplomacy a bit at the expense of conquest and space.	
`MaxWarDistantPowerRatio=70` for Napoleon. Napoleonic France fought few off-shore wars and even ceded Louisiana to the US.	Was 100. A preference for naval war. Napoleon is tied for the highest value, while his MaxWarNearbyPowerRatio (land wars) is only moderate.
`MaxWarMinAdjacentLandPercent=2` for Charlemagne and Cyrus. Charlemagne extended his borders gradually in all directions; Cyrus conquered several empires that weren't exactly at his doorstep.	Was 1 and 3 respectively.
`MaxWarRand=150` for Willem, i.e. now a bit less willing to start "total" wars. In part, because wars of conquest seem out of character for the small Netherlands; in part, because he's doing a bit too well in AdvCiv games in my experience (and more so than other Financial leaders). Also shifted his victory weight a bit from Conquest to Culture and made him a little less interested in dogpile wars. And increased `NoWarAttitude` when Pleased to 90.	Was 100. Was 80, tied for the lowest value with Alexander, Louis, Mao, Montezuma, Qin, Victoria. 90 is the same as the other 19 leaders who attack when Pleased
Increased Roosevelt's `NoWarAttitudeProb` at Pleased to 100 and at Cautious to 70 but increased his `BuildUnitProb` to 25. Decreased `NoWarAttitudeProb` for Brennus at Cautious to 50 and for Darius at Pleased to 90.	Was 90 at Pleased, 50 at Cautious, 20 BuildUnitProb. Was 70 (Brennus, Cautious) and 100 (Darius, Pleased).
Negotiating a trade embargo against Gandhi results in a -1 relations penalty. Agreeing to an embargo against him still carries no penalty. Gandhi doesn't mind being voted against.	Embargoes against Gandhi carry no penalty for either side. Voting against him carries the usual -2 penalty.
Suryvarman's favorite religion is Hinduism, primary flavor is Religion and fav. civic Caste System. He was a Hindu and built large temples. (Some later Khmer kings were Buddhists.)	Buddhism. Primary Gold, secondary Culture. Gandhi is the only Hindu. Fav. civic: Organized Religion; Wang Kon is the only Caste System proponent.
Overhauled Unit AI weight modifiers and reduced their impact to at most +50% per Unit AI type. And two small accompanying changes to `CvPlayerAI::AI_unitValue`.	Each leader (except Hatshepsut) has 100% weight bonus for one particular Unit AI type, namely *Worker* (Pacal, Frederick, Gandhi), *Explorer* (Mansa Musa), *Attack* (Alexander, Brennus, Cyrus, Hannibal, Justinian, Montezuma, Peter), *Attack City* (Augustus, Chalemagne,
Also overhauled AI improvement weight modifiers and increased their impact to up to 30% per improvement type.	Kublai Khan, Louis, Suleiman), *Counter* (Catherine, Julius, Napoleon, Roosevelt, Saladin, Wang Kon, Willem), *City Counter* (Darius, Huayna Capac, Lincoln, Tokugawa), *City Defense* (Churchill, Gilgamesh, Hammurabi, Sitting Bull,
See also: [131](#) makes minor changes to `AI_bestUnit`, and `AI_getImprovementValue` (both in `CvCityAI.cpp`), the functions that apply the weight modifiers. Civ 4 Reimagined has also overhauled unit and	Zara Yaqob), *Reserve* (Ashoka, Ramesses, Suryavarman, Washington), *Pillage* (Boudica, Genghis Khan, Mao, Shaka),

improvement weight modifiers: [Git commit](#)	Collateral (Bismarck, Mehmed, Qin), *Attack Sea* (Isabella, Pericles, Peter), *Reserve Sea* (Victoria), *Assault Sea* (Ragnar), *Explore Sea* (Elizabeth, Joao), *ICBM* (De Gaulle). 31 leaders have 10% weight bonuses for one or two improvements, namely *Farm* (Bismarck, De Gaulle, Shaka), *Farm & Windmill* (Brennus, Boudica, Churchill, Elizabeth, Willem, Napoleon, Peter, Pericles, Cyrus, Hammurabi, Mehmed, Suleiman, Isabella, Joao, Pacal, Washington), *Cottage..Town* (Darius, Frederick, Huayna Capac, Ragnar, Victoria, Wang Kon), *Cottage..Town & Windmill* (Gandhi, Hannibal, Lincoln, Mansa Musa), *Workshop & Watermill* (Mao, Stalin).
Rationale	Roosevelt change: 90/50 NoWarAttitudeProb is the same as the Mongol leaders or Napoleon. Some peaceful leaders like Mansa Musa and Elizabeth also have it, but along with a low BuildUnitProb. This makes for leaders that are ruthless but powerless. This doesn't fit for Roosevelt at all. He's the leader most representative of the present-day America that doesn't invade partners, but has a big military. Changed Brennus and Darius to keep the overall balance. I don't know if that balance is exactly right, but, if anything, there are too many leaders that don't normally attack at Pleased.
	Suryavarman's fav. civic: Asking others to change into Organized Religion (or Theocracy, Pacifism) generally doesn't make much sense unless the state religions happen to match. This was especially annoying in Sury's case because his change-civics requests apply a -2 relations penalty when rebuked.
	Unit AI weights: Doubling the weight seems extreme, and, for most leaders, a preference for a particular Unit AI type doesn't even make much sense. See `Civ4LeaderHeadInfos.xml` for the new weights and some handwave rationales. I've tried to avoid reinforcing the AI type(s) of the unique units because the AI already trains these more frequently than the unit stats would justify. About Worker weights, note that Workers are rarely trained through `AI_bestUnit`, so these have very little impact. This may also apply to other unusual bonuses (e.g. Spy).
	Improvement weights: I can't make much sense of the BtS weights. Looks like the AI didn't build enough Windmills and the developers decided to amend this through weights for all civs that have ever built windmills. This reason should be obsolete because of K-Mod changes in the DLL, and 10% is very little (it said 20% in XML, but the DLL halved that). I've given most leaders two 20% bonuses (no longer halved), and some a single bonus of 30%. The impact of these bonuses still seems very small, perhaps too small. The choice of the improvement is now vaguely based on the leaders' historical policies. I didn't take into account whether an improvement supports a leader's traits, favorite civics or playstyle; the DLL can handle that, and I don't want to make the playstyles (even) more predetermined.
Tbd.	Considering (low-key) changes to make certain leaders more distinct, improve historicity and to increase the number of warlike leaders. Under consideration for becoming more dangerous: Augustus, Brennus, Cyrus, Darius, Churchill, Frederick, Hammurabi, Justinian, Ramesses.
	Willem is still too aggressive. May need a thorough overhaul to become a small civ interested in culture and only slightly feisty.
Increased cap for the relations penalty from shared borders to 3 for Mao, Stalin and Darius, and decreased the cap to 3 for Qin and to 2 for Roosevelt.	2 for Mao, Stalin and Darius, 4 for Qin, 3 for Roosevelt. The cap is between 1 (only Willem has this value) and 4 (most of the typical warmongers).
Rationale	Modern China has plenty of border problems with pretty much all its neighbors. In Mao's time, border tensions with the Soviet Union escalated. Stalin was impossible to get along with for his western neighbors. I'd set both caps to 4, but that change would

	be a bit drastic. Both Persian kings invaded most of their neighbors, but let's keep Cyrus somewhat peaceable for now. To counter the increased caps, I've reduced Roosevelt (I don't see why he should be particularly sensitive about border tiles) and Qin. The Qin dynasty secured its northern border with the Wall, but that doesn't quite justify a cap of 4. And it's strange to give Qin a higher cap than Mao.
Partially revised the `CONTACT_DEMAND_TRIBUTE` divisors.	A high divisor means that the AI leader is unlikely to demand tribute.
Ragnar, Mehmed, Shaka, Kublai Khan and Brennus set to 25.	All five 1000.
Montezuma set to 50.	25
Charlemagne, Wang Kon, Hammurabi, Gilgamesh, Suryavarman, Joao, Julius Caesar, Napoleon and Genghis Khan set to 250.	1000, 1000, 500, 500, 500, 50, 50, 25, 25 respectively.
Louis set to 500.	50
Catherine set to 750.	50
De Gaulle, Sitting Bull and Isabella set to 1000.	25, 25 and 50.
Rationale	For change 079 (bragging about units based on the demand-tribute divisor), I'd like early warmongers to have a low divisor so that they blab. That said, the divisors also need to make sense historically, which, to me, wasn't the case at all in most of the cases that I've changed. In that regard, it should matter e.g. whether the ruler or the dynasty he or she represents had exacted tribute or similar payments (though tax-like tribute from provinces or satrapies shouldn't count) and if tribute was customary in their cultural area. It can make sense to let leaders that represent non-expansionistic powers ask for tribute frequently – they prefer tribute over conquest, and vice versa. However, for modern peaceful leaders (e.g. Lincoln; not changed), it makes more sense to let them start wars (for reasons that are somehow principled) than to let them extort payments.
Tbd.	I'm also not happy with most of the 33 leaders that I haven't changed. Should address those as part of a comprehensive leader overhaul.
See also	104m adjusts the probability for an AI tribute demand based on war utility.
Gave Gandhi the highest no-war percentage at Annoyed attitude, namely 50, the second highest at Cautious (80 after Sitting Bull who has 90) and made him the only leader with a no-war percentage at Furious: 20. Decreased Sitting Bull's percentage at Annoyed to 40 and Lincoln's to 30. Also decreased Justinian's percentage at Cautious to 70.	Pleased/Cautious/Annoyed/Furious Gandhi: 100/70/30/0 Sitting Bull: 100/90/50/0 Lincoln: 100/80/40/0 Justinian: 100/80/30/0 For reference: Ashoka, Augustus, Churchill, Cyrus and several others: 100/70/20/0
Rationale	A faithful representation of Gandhi should be 100/100/100/100. Not sure about the last two numbers; Gandhi did have a pragmatic side. Cf. Gandhi on Indian involvement in WW2 (Wikipedia). More importantly, players tend to assume that Gandhi acts very peacefully in the game. In BtS and even K-Mod, his high `WarRand` values make it very unlikely for him to start wars and his military budget, the only factor that can dynamically lower the `WarRand` values in BtS/K-Mod, is virtually always small. UWAI introduces a bunch of other opportunistic factors, making the `WarRand` values far less relevant in situations when starting a war is highly beneficial. I didn't want to increase the no-war percentages overall – if anything, I feel that they're

	too high overall –, so I shuffled them around: Gandhi takes his Annoyed percentage from Sitting Bull, Sitting Bull from Lincoln, Lincoln from Gandhi. Gandhi swaps his Cautious percentage with Justinian. Sitting Bull can't be more peace-loving than Gandhi overall. I did preserve his especially high reluctance at Cautious attitude. Justinian's unusually high no-war percentages don't seem to have any historical basis. It's OK that he's not supposed to fight civs that share his religion, but, given his very high `SameReligionAttitude ChangeLimit`, Cautious attitude shouldn't affect that behavior. Lincoln: want the 40 for Sitting Bull. 30 is still unusually high.

005b	GP names assigned chronologically

AdvCiv	*BtS*
GP names are assigned in the order they're listed in `Civ4UnitInfo.xml`, i.e. roughly ordered by date of birth; no more Ancient Heisenberg. About every second (based on the number of civs in the game) name is skipped at random, so that GP names aren't the same in every game.	When a GP is born, the name is chosen uniformly at random from among the GP with matching type. Heisenberg is just as likely to be the first Great Scientist as Socrates.
There is an additional offset when starting in the Medieval era or later, i.e. early names are skipped.	
Corrected a few misspelled GP names, e.g. "Frank" Kafka.	
Replaced the two Great Generals that also appear as civ leaders with two new ones (Charlemagne→ Zizka; Boudica→ Hai Ba Trung).	
Names of playtesters replaced with actual spies from the 20ᵗʰ century.	The last 10 names for Great Spies are the real names or nicknames of BtS playtesters.
"Malian" as the ethnonym of Mali	"Malinese"

Credits	The new spy names are taken from a list that etiennefd (Steb on CFC) compiled for the RFC: Dawn of Civilization mod. This CFC post by Phil Bowles made me aware that "Malinese" has apparently only been popularized by the Civ series.
See also	038 picks historians that somewhat match the type of civ ranking (e.g. Machiavelli for the "most powerful civilizations").
Tbd.	Perhaps separate lists of GP names for every civ. DoC should have lists for most civs. Though I'd like the following better for AdvCiv: Assign a list of GP names to each technology and pick a name based on the current bulb tech when a GP is born. Use the BtS list as a fallback (and for Great Generals). May have to add a separate list for each GP type to each entry in Civ4TechInfo for this. Another potential name for a Great Spy (from Realism Invictus): Thomas Edward Lawrence

005c	City ruins bear the name of the former city

Help text shows the name of the most recent city in a tile with city ruins. Can't pillage city ruins, meaning they can only be removed by building an improvement on top.	The name of the former city isn't recorded anywhere. Can pillage city ruins (0 gold).

Rationale	For flavor. No pillaging because the name of a destroyed city could have strategic importance (albeit very little), and I wouldn't want players to destroy ruins in order to monopolize information (in multiplayer). Also, while Sennacherib boasted to have "removed the dust of Babylon for presents to be sent to the most distant peoples," the idea of pillaging ruins doesn't immediately make sense.
005d	Spurious/ misattributed tech quotes corrected or marked as "ascribed to ..." (only English and German)
Credits	CFC discussion about Civ6 tech quotes: link; offshoot about BtS: link CFC user Steb pointed out a few more in the Dawn of Civilization subforum: link
005e	German text: Changed the name of Galley from "Triere" to "Galeere" and the name of Trireme from "Trireme" to "Triere" because "Triere" and "Trireme" are synonyms. The names are also problematic in English: A trireme is a special kind of galley. The Warlords Civilopedia text says that the Galley unit is supposed to be a pentekonter. Apparently, they didn't want to give Galley that rather obscure name.

005f	City art style based on highest culture	
The art style of a city's 3D models is that of the civ with the highest tile culture in the city.		The city owner's art style is used.
Rationale	The style of the buildings shouldn't immediately change upon conquest. (It kind of did in the New World, but in these cases much of the former population and its culture was also gone.) It's nice to have more traces of the past on the map. Disabled this change again after realizing that there is no way to make the same change for Cottages, Hamlets etc. Looks like the code in the EXE checks the owner of the improved tile, calls `CvPlayer::getArtStyleType` on the owner and uses the returned `ArtStyle`.	
Config	XML switch in `GlobalDefines_advc`	
Credits	Idea from Xyth's History Rewritten v1.25	

005g	City name tweaks
Tbd.	All the city lists could use an overhaul: Higher priority for ancient capitals; avoid high priority for cities that were very close to each other; include more remote cities, maybe even some in client states that don't appear in the game. Mods like Realism Invictus and DoC could be of help.

AdvCiv	*BtS*
Angkor Wat removed from the Khmer city list.	It's in position 4. Yasodharapura is in position 1.

Rationale	The temple was in Yasodharapura, also known as Angkor ("capital") but not as Angkor Wat ("capital temple"). The temple district could be seen as its own city but isn't commonly seen that way, and having a wonder and a city of the same name is awkward.
Tbd.	Chichen Itza: Should rename the wonder in this case (Temple of Kukulcan).

005h	Changes to Actual Quotes diplo text

AdvCiv	*Actual Quotes*
Rename the Actual Quotes game text file to `Civ4ActualQuotesText.xml` and include only	The Actual Quotes game text file is a copy of `CIV4DiplomacyText.xml` from Vanilla Civ 4.

text that differs from Vanilla Civ 4. Added in-line comments about changes to game text that isn't leader-specific.	Many texts have been modified (English only), many haven't been. It's clear enough that most of the leader-specific texts have been replaced, but there are also changes to generic texts.
Rationale	Having everything in one place has advantages, but that's not the case anyway because Warlords and the BtS expansion have their own diplomacy text files; and being able to review the changes made by Actual Quotes seems more important.
Restored a few whimsical texts from Vanilla Civ 4 that Actual Quotes had replaced.	Willowmound seems to have gone after highly colloquial texts and especially texts that make the AI leaders appear as opponents in a boardgame as opposed to historical figures. Texts that belittle the human player also have been targeted (perhaps, in part, because they don't fit well in situations when the human player is doing far very well).
Rationale	I get why those lighthearted texts were disabled, but some them are kind of classic, especially some of the insulting ones. They don't mix well with the Actual Quotes texts, but there are plenty of old (generic) texts that don't fit well with the new texts either. And some of the Actual Quotes replacements for the "offending" texts were pretty bland. On that note, many of the leader-specific texts should've been paraphrased more loosely in my opinion; they're too stilted.
Added one text for a declaration of war (DoW) by Gandhi and one by Alexander.	Actual Quotes pretty much only replaces Vanilla text, and there are no leader-specific texts in Vanilla for DoW.
Rationale	I happened to come across some quotes that seemed suitable for those two leaders. Could be an inspiration for someone to add more leader-specific DoW texts.

006	Assertions
Disabled a few assertions that are supposed to be rare and were still under investigation by earlier modders, and are probably false positives or mostly harmless and difficult to resolve.	
006b	Enabled assertions in CvXMLLoadUtility that warn when an XML tag expected by the DLL isn't found in XML – unless a default value is passed to `CvXMLLoadUtility::GetChildXmlValByName`.
Rationale	If the tag is defined in the schema, then the EXE will warn about it anyway, but the new assertions should help when a tag is also missing from the schema and, in particular, when the tag is mistyped in `CvInfos::read(CvXMLLoadUtility*)`. And they warn about missing default values for non-mandatory tags – although a default of 0, `false` or an empty string as used by BtS is usually fine. I pass default values to `GetChildXmlValByName` where they are now necessary.
See also	No assertion triggers when a default value is set through xmldefault.
Tbd.	Can I somehow warn about tags that are entirely unused?
006c	Define assertion and profiling macros as `(void)0` when disabled.

Rationale	Mainly to avoid errors like this:
	``
	if(...)
	   FAssert(...)
	foo();
	``

	If `FAssert` is entirely removed (as in BtS), then the `foo` call will move into the `if` branch without any compiler error. With my change, macro calls have to be terminated with a semicolon.
006d	Warn when reading an XML element with an empty value (e.g. `<iMoves></iMoves>`)
Credits	Adopted from [More Naval AI](#)
006e	Show current XML file in error messages only if the error occurs while loading an XML file.
Credits	A couple of lines from [these](#) [two](#) MNAI commits.
See also	[advc.rh](#): Adopted a similar (redundant?) change from rheinig's mod.
006f	Show the name of the enclosing function in in `FAssert` popups through the `__FUNCTION__` macro; remove the (now redundant) function name parameter from `FAssertBounds` (formerly named `FASSERT_BOUNDS`).
Credits	From [Caveman2Cosmos](#) (billw2015)
Tbd.	Would be nice to include stack trace as well; that would arguably require some library; C2C uses [Stack Walker](#).
See also	[advc.enum](#) introduces a macro for asserting the bounds of info enum types.
006g	Show a failed assertion instead of a windows error message (`ErrorMessage`) when XML loading fails while debugging.
Rationale	The assert popup is much more helpful for debugging. Windows error messages are still shown by the DLL-internal profiler (`CvGameCoreDLL.cpp`). I haven't changed those because one doesn't normally debug while profiling
006h	When an assertion fails in a debug build, the condition is evaluated for a second time.
Rationale	So that the user can immediately step through the evaluation of the condition. It's not always obvious why it evaluates to `false`. I don't think there's any downside to this. Failed assertions are rare, so performance isn't consideration. Assertion conditions must never have side-effects. And just pressing F10 will skip to the line after the `FAssert`/`FAssertMsg` call as before.
006i	Macro `FErrorMsg` instead of `FAssertMsg(false, ...)`
Credits	From Caveman2Cosmos: [Git commit](#)
006j	After storing a simple integer element from XML in an enum variable, the limits of the enum type are asserted.
See also	Based on the enum traits code written for [advc.enum](#).
006k	Warn when loading a CvInfo element from XML for a second time despite modular loading being disabled.
Rationale	Not going to leak memory, but probably not intended by the XML modder.

007	Changes to info shown in Debug mode; logging

- No confirmation needed for entering WorldBuilder if already in Debug mode.
- Red circles from BBAI only shown if Show-Yields view was enabled before entering Debug mode. (Can also toggle the circles by toggling Show-Yields and entering and leaving a city

screen.)
- Military Advisor shows all foreign units.
- Privateer owner revealed.
- Alt while hovering over a leader portrait in Debug mode shows the leader's id (slot number).
- Alt+Ctrl on an owned tile shows the AI resource trade counter (which is divided by 50 to yield the "appreciate the years" relations bonus).
- Alt+Shift on an owned land tile shows the number of AI Workers needed in the tile's area.
- Can no longer show Info, Victory, Espionage and Foreign Advisor screen for Barbarians. These screens aren't helpful and some had been leading to failed assertions.
- The Wonders tab (Info screen) shows all wonders in Debug mode – unless a player other than the active player is selected from the Debug menu. (The menu would be moot if all info was always shown regardless of the selected player.)
- When Alt is held down on a tile, the tile's found values are recomputed ignoring overlap with planned city sites. This value is shown first, then the cached value (planned sites taken into account). (BtS shows the cached value first, then a recomputed value that normally equals the cached value and then then a value computed as if the map was empty, marked with an "s" for "starting location". AdvCiv does not compute this "s"-value.)
- Alt-text on the scoreboard shows culture info only when Alt+Ctrl is held or when a civ is close to a culture victory.
- Don't show espionage help for rival spies; e.g. no "unit has moved/ can't conduct mission" message in red letters.
- Some changes to the information shown by the contact-player widget on the scoreboard about unmet players when in Debug mode. (`CvDLLWidgetData::parseContactCivHelp`)
- Add a `bForce` parameter to `CvPlayerAI::AI_isDoStrategy` that CvGameTextMgr and CvDLLWidgetData can use in order to ignore `CvPlayer::isHuman`. So that the scoreboard in Debug mode will show human strategies.
- No time stamps in the BBAI log. (I've never needed them for anything and they're very annoying when comparing logs in order to debug synchronization problems.
- Ctrl+Alt while hovering on a capital shows AI tech values (was Ctrl+Shift on any owned tile)
- Need to hold down Shift in order to see stack compare values along with combat odds.

See also	[001d](#) fixes some Debug mode bugs in various Advisor screens. [085](#) shows a score breakdown when Ctrl is held in Debug mode while hovering over a score value on the scoreboard. [058](#) deals with concealed player identities – and makes sure that they're not concealed in Debug mode.

XML switch `PER_PLAYER_MESSAGE_CONTROL_LOG` for creating multiple `MPLog.txt` when testing multiplayer games on a single PC. These will have to be deleted manually. In networked multiplayer games, the id of the active player always gets appended to the name of the BBAI log (no XML switch for that one).

Procedure: Set `LoggingEnabled=1`, `OverwriteLogs=1`, and `RandLog=1` in `CivilizationIV.ini`. Will probably also have to set `MessageLog=1` and `SynchLog=1`. Or rather, put these in a copy of `CivilizationIV.ini` (say, `MPDebug.ini`), set `Mod = Mods\AdvCiv` and `FullScreen = 0` in the copy and place it in the same folder as `Civ4BeyondSword.exe`; then create a shortcut targeting `"C:\Program Files (x86)\Sid Meier's Civilization 4\Beyond the Sword\Civ4BeyondSword.exe" ini=advcMP.ini multiple`, where `multiple` is for allowing multiple instances of the BtS process. Launch the game twice through that shortcut, create a Direct IP game named `chipotle` (see 135c) with one process and join it with the other by connecting to `127.0.0.1`.

Rationale	Need an `MPLog.txt` from each player in order to debug OOS errors. The BBAI log can also be helpful.
See also	[001n](#) fixes OOS errors [135c](#) allows debug tools in multiplayer and refactors `CvGameTextMgr::setPlotHelp`.

	Disabled some log output from `CvEventManager.py`. City growth, border expansion and saves are now only logged if the respective `__LOG` variables in the CvEventManager constructor are set.
Rationale	Probably an oversight by the (Vanilla) Civ 4 developers. Most of the other log output was already tied to `__LOG` variables. (Savegames were probably being logged on purpose, but that leads to a lot of uninteresting log output when autosaving every turn.)

A short guide about the various logfiles:	
`LoggingEnabled` (`CivilizationIV.ini`)	Enables several log files, most importantly `PythonErr.log` and `PythonDbg.log`.
	Use `CvUtil.pyPrint`, `print`, or `BugUtil.debug` to write to `PythonDbg.log`. In the DLL, `CvDLLUtilityIFaceBase::logMsg(logFileName, …)` can be used.
	The BUG Python logs have an extra switch on the System tab of the BUG menu.
	LoggingEnabled also enables: - `audio.log` - `xml.log` - `network*.txt` - `app.log` (mostly seems to record the Civ 4 window gaining or losing system focus) - `resmgr.log` (failures to load `.dds` graphics; models and animations too?) - `initmgr.log` (apparently for profiling game initialization), - `LSystem.log` (some type of graphics warnings) - `combat.txt` (K-Mod – requires `LOG_COMBAT_OUTCOMES` to be defined in `CvUnit.cpp` in addition).
	Disabling `LoggingEnabled` doesn't stop all messages written to `xml.log`, I guess because of the timing inside the EXE. Also probably by accident, some messages still appear in the `network-*.txt` logs.
	There seems to be no way to check the value of `LoggingEnabled` programmatically; only the EXE knows. That said, the DLL can enable or disable `LoggingEnabled` (or any other key in `CivilizationIV.ini`) through `gDLL->ChangeINIKeyValue`. One can probably also use `CvDLLIniParserIFaceBase` to re-parse `CivilizationIV.ini`.
`PythonErr2.log`	Can't be disabled, seems to be used only by compiled Python code from the Warlords expansion. Basically never of interest.
`SynchLog` (`CivilizationIV.ini`)	Mainly seems to write "*DBG: SYNCLOG: All Checked In*" messages to `network-*.txt`. That happens exclusively in the EXE; it doesn't look like the DLL can write to that logfile.
	To check whether `SynchLog` is enabled, use `CvGlobals::isSynchLogging`. I think `LoggingEnabled` is required in addition.
`MessageLog` (`CivilizationIV.ini`)	Enables the "message control log" `MPLog.txt`. It's mostly a combat log (less detailed than the in-game log and `combat.txt`).
	Use `CvDLLUtilityIFaceBase::messageControlLog` to write to `MPLog.txt`. 003 wraps a class CvDLLLogger around that function.
	Use `CvGlobals::isLogging` to check if the log is enabled. Note that this isn't the same as `LoggingEnabled` (see above), which is not a prerequisite

	here. The EXE shows a warning on the Staging Room screen if a player has the message control log enabled. advc.mapstat writes to the message control log if `LOG_MAP_STATS` is set in `GlobalDefines_devel.xml`. If `PER_PLAYER_MESSAGE_CONTROL_LOG` is set in `GlobalDefines_devel.xml`, then multiple files `MPLog*.log` are created in multiplayer games. Cf. 135c.
OOS log	When the game recognizes an out-of-sync error in a network game and `MessageLog` is enabled, `OOSLogger.py` creates a file `OOSLog*.txt` and writes all data that the OOS checksum is computed from to that file. Comparing the message logs of all players can give a clue about the cause of an OOS error. The log is no help, however, if only one player has `MessageLog` enabled. Realistically, it's only useful for OOS debugging with multiple program instances on a single machine. Cf. kekm.27.
`RandLog` (`CivilizationIV.ini`)	Logs all pseudo-random numbers generated by an instance of CvRandom. Writes to `MPLog.txt`, so `MessageLog` needs to be enabled in addition. (To find the combat log messages in between the PRNG messages, one can search for "kombat".) Through advc.007c, log output of the global non-synchronized RNG `CvGlobals::m_asyncRand` are written to `ASyncRand.log` instead of `MPLog.txt`. I.e. MessageLog.txt isn't required for this RNG. Use `CvGlobals::isRandLogging` to check if `RandLog` is enabled.
`LOG_AI` (`BBAILog.h`)	Need to recompile to toggle this log. Pretty comprehensive AI logging (with some blank areas). Thus also useful for OOS debugging. Writes to `BBAI.log`. Requires `LoggingEnabled=1`. Use `logBBAI` to write to `BBAI.log`. Such calls should be preceded by a `g...LogLevel` check ("g" for global), otherwise, the call will take up time even if the BBAI log is disabled. (In `CitySiteEvaluator.cpp`, I've wrapped that check together with the `logBBAI` call into a macro.)
UWAI reports (`GlobalDefines_` `devel.xml`)	Reports about AI war planning by the UWAI component; written to files named `UWAI*.log` once per team turn if the game turn number is divisible by `REPORT_INTERVAL` in `GlobalDefines_devel.xml`. Additionally requires `LoggingEnabled` and – to avoid cheating – `MessageLogging` in multiplayer games. Use `WarAndPeaceReport::log` to write to the current game turn's `UWAI*.log`. Use `WarAndPeaceReport::isSilent` or `isMute` to check whether the report object is enabled. If not silent, then temporary mute status can be toggled through `WarAndPeaceReport::mute(bool)`. Unfortunately, re-launching Civ 4 doesn't cause old reports to be overwritten; will append instead. The AdvCiv makefile has a `YOURLOGS` variable though that will cause all `.log` files to be deleted after recompilation.
`StartingPos.log`	Requires recompilation to toggle the `SPI_LOG` switch in `StartingPositionIteration.cpp`. `LoggingEnabled=1` is also required. The log documents the steps taken by the starting position iteration

	algorithm; see <u>027</u>.
Profiler log	The DLL-internal profiler, if enabled, writes to `IFP log.txt`. See <u>003o</u> and `Makefile.project` about that.
`ThemeParseLog.txt`	Gets created in the BtS install directory!
VS console	When launched from within Visual Studio, debug builds write some messages to the "Output" window of Visual Studio. The DLL can write such messages through `OutputDebugString`, defined in `CvGameCoreDLL.h`. Messages written by the EXE might be helpful for diagnosing program instability; though the "first-class exceptions" are generally no cause for concern. If the Visual Studio debug heap is enabled, then messages about memory errors are also written to the Output window when Civ has exited.

mapstat	Log file with statistics about a generated map	
AdvCiv		*BtS*
If `LOG_MAP_STATS` is set in `GlobalDefines_devel.xml`, and `MessageLog` enabled in `CivilizationIV.ini`, then, each time a map is generated, some statistics like land/ sea ratio, terrain, elevation and resource frequencies, resources per player are written to `MPLog.txt`.		There is a "Map Details" screen available in Debug mode through Shift+F2.
Rationale	The Map Details screen is difficult to read and lacks terrain info.	

007b	Cheat and debug shortcuts disabled unless in Debug mode	
AdvCiv		*K-Mod*
Unless in Debug mode, only the shortcuts Ctrl+Z (enter Debug mode), Ctrl+D (menu for graphics debugging), Ctrl+U (unit graphics debugging), Ctrl+Alt+T (reset game text), Ctrl+Alt+A (audio viewer), Ctrl+Alt+L (reset city layout), Ctrl+Shift+P (change civ), Ctrl+Shift+L, Alt+[Shift+]Z (switch player) and Ctrl+Shift+X/B (AI Auto Play) are enabled. (There could be others that I'm unaware of.)		Numerous shortcuts (<u>list</u> on GameSpot.com) that require only "chipotle" to be entered in `CivilizationIV.ini`; Debug mode isn't checked. Some of these are easy to trigger by accident, in particular Ctrl+Shift-+Left Click (Object Placer menu) and Plus key while hovering over a city (immediately finishes the city's production).
Ctrl+Alt+R disabled entirely, though a message "Reloaded Art Defines" (sent from the EXE) still appears – along with a warning from AdvCiv code that it's actually disabled.		Ctrl+Alt+R crashes the game, also in BtS; probably one of the expansions has broken it.
Rationale	If players actually want to use these shortcuts for cheating a little, then it could be annoying that Debug mode is required because Debug mode reveals the map, which can be a spoiler (and so does WorldBuilder). Still, for development purposes, I need to have Debug mode at hand, so I need the cheat mode ("chipotle"), but I don't want to run into other cheat commands, perhaps without even noticing.	
See also	Chapter on <u>all-AI games</u>: The spectator mode requires Debug mode, which in turn requires cheats to be enabled, so regular players may end up leaving cheats enabled permanently. Don't want these players to stumble onto cheat commands.	

Tbd.	Could add an override in GlobalDefines to allow cheat commands without Debug mode.
	Perhaps the reloading of Art Defines wouldn't be so difficult to fix.
	Would like to block the reset/reload shortcuts when not in Debug mode. `onKbdEvent` in `BUG`/`CvEventManager` doesn't help – apparently, the key press gets processed by the EXE before that handler is called. The only way I see: During initialization, store `gDLL->getChtLvl()` in a (serialized) variable at CvGame and set `gDLL->setChtLvl(0)`. This will disable all cheats. When Ctrl+Z is pressed (I think BUG handles that already) and not currently in Debug mode, ask CvGame what the cheat level is supposed to be, and, if it's greater than 0, call `gDLL->setChtLvl(1)` just before `CvGame::toggleDebugMode`. When Ctrl+Z is pressed in Debug mode, call `gDLL->setChtLvl(0)` after `toggleDebugMode`. Will have to integrate this with [135c](), which allows Debug mode in multiplayer.
	Would also like to block Ctrl+Alt+F4 (kills the process – or simply exits to desktop without confirmation; not sure). This might be an OS thing, but the fact that it doesn't work in the main menu suggests to me that it's implemented in the EXE.

Enabled some debug tools that the BUG mod had (accidentally I guess) made unavailable. Now they require Debug mode (probably only required cheat mode in BtS):

```
Shift+T:         Award tech or gold.
Shift+Ctrl+W:    View wonder movie
Shift+] on unit: Heal 10 HP
Shift+[ on unit: Damage 10 HP
Shift+F1:        View replay
Shift+F2:        "Debug Info" screen with various per-civ statistics
Shift+F3:        View Dan Quayle screen
Shift+F4:        View UN victory screen
```

Each cheat command is triggered only by one combination of modifier keys; e.g. Shift+Ctrl+T will not open the tech/gold menu.

Rationale	Reduce the risk of clashing shortcuts, i.e. one event handler pre-empting another.	
AdvCiv		*BUG*
Removed handlers for keyboard inputs from `CvEventManager.py`.		`BUGEventManager.py` handles keyboard inputs.
Rationale	Removal of dead code.	

007c	Added macros for RNG calls that make the logging side less tedious – by simply writing the current function name and line number into the log. Replaced all calls to `CvGame::m_sorenRand` and `CvGame::m_mapRand` with those macros.
	Output of the global non-synchronized RNG is written to `ASyncRand.log` instead of `MPLog.txt`. (If `RandLog` is enabled in `CivilizationIV.ini`.)
Rationale	I'm not sure if a log file for the non-synchronized RNG is helpful at all. In any case, having synchronized and non-synchronized numbers in one place is potentially confusing and makes it harder to debug OOS errors. The Firaxis developers had sometimes added "*ASYNC*" to messages from the non-synchronized RNG, to make them stand out I guess. This is no longer necessary, and removing those suffixes has allowed me to simplify `CvPlayerAI::AI_techValue` a bit.

Tbd.	Might be better not to use any random numbers in `CvPlayerAI::AI_techValue` when the function is called asynchronously, i.e. when recommending techs to a human player.
See also	advc.fract: The RNG macros , apart from integer percentages and limits, also support the ScaledNum type.

008	Changes to Civilopedia content and hints
I've updated some content that is no longer accurate. For some Game Concept pages, I left the content alone, and merely added a warning at the top that the info could be outdated.

I've only changed the English and German version. I haven't updated the German list of keyboard shortcuts (004y); it's still as in the BUG version of Sevopedia. Other than that, I think the German translations in Civilopedia are complete.

Config	The modified bits are in a separate file called `CIV4GameText_advc_pedia.xml`. I've also moved the one text changed by K-Mod into that file (from `Civ4GameText_K-Mod.xml`).
Tbd.	Not all the strategy advice is up to date. Especially the tech and building advice is a lot of work to maintain, and much of it is redundant. The tech advice is only shown when Sid's Tips are enabled. Would be best to somehow disable these texts completely for anything past the Ancient era (simply check the tech's era along with each check for the player option); the early-game texts actually do contain some advice.
See also	004y: Changes to Civilopedia formatting 033 shows the Privateer's plunder ability in Civilopedia 179 shows the production bonus of the Apostolic Palace in Civilopedia
008a	Hide abilities in Civilopedia when they don't apply
Rationale	To unclutter the Civilopedia
See also	004w unclutters the help texts for buildings on the city screen.

AdvCiv	*BtS*
Showing the "Can be built on X era and earlier starts" restriction only when X is smaller than the current game's start era, i.e. when the restriction applies to the current game. When the Civilopedia is accessed from the main menu, the restriction is shown when X is not equal to the wonder's tech era plus 1.	The restriction is shown for all buildings that have one regardless of how Civilopedia is accessed and regardless of the current game's start era. For most wonders, X equals the wonder's tech era plus 1.
Shwedagon Paya not available on Renaissance start and Statue of Zeus available on Classical start.	Both wonders were added in BtS. Shwed is available on Renaissance start and earlier and Zeus only on Ancient start.
Versailles not available on Industrial start.	Industrial start or earlier.

Rationale	Most wonders become unavailable when starting two eras after their tech era; spell out only these exceptional cases (e.g. most Ancient wonders are available when starting in the Classical era, but Stonehenge isn't). The changes to individual wonders reduce the number of exceptions that the Civilopedia needs to mention.
See also	310 changes the start era restriction of the Great Wall
Tbd.	Civilopedia should list the free buildings from later-era start along with the descriptions

	of the later-era start settings.
	Perhaps prevent civs from constructing a wonder once they reach the threshold era – regardless of the game's start era. This could make wonders scarcer in games with few civs.

Show required victory condition ("such-and-such victory must be enabled") only when accessing Civilopedia from the main menu or when the victory condition is disabled in the current game.	
008b	(unassigned)
008c	Updated some of the hints that are shown while loading savegames (only English and German)
008d	New hints about AdvCiv changes and a few K-Mod changes that weren't previously covered. Gave all hints about modded content a higher probability of being shown than the hints from Vanilla and BtS (Warlords didn't add any). Tried to mimic the tone of the original hints. Only in English and German.
Config	Increased the probability by duplicating entries in `CIV4Hints.xml`. Separate file for the new hints: `CIV4GameText_advc_hints.xml`.
008e	Changes to the names of wonders and projects
No wonder or project name starts with "The" or any other article (translations).	Totally inconsistent; e.g. "Angkor Wat", but "The Taj Mahal".
Rationale	Makes it easier to find wonders in Civilopedia and on the Wonders tab. Just "Pyramids" is a little strange, but most of the names work fine without an article. Could make it "Great Pyramids" (joining Great Lighthouse, Great Wall and Great Library), but I guess "Pyramids" is OK.
Config	Separate file `CIV4GameText_advc_wonders.xml`. Further changes to wonder names will have to be made there. Can delete the file to undo all name changes.
See also	Caveman to Cosmos also does this (but I haven't checked how they've implemented it). SourceForge [revision](#)

009	Setup of configuration files: BUG settings, GlobalDefines, BUG and BBAI help files	
See also	[002b](#) also deals with file paths	
AdvCiv		*K-Mod*
Included the BUG help files in three languages (no French and Spanish translation seem to exist).		No BUG help files included; clicking on "BUG Mod Help" in the BUG options menu results in an error message.
Rationale	The help files also include developer credits; gotta have those.	
Tbd.	Would be nice to use the [SourceForge URL](#) that `BugHelp.py` tries to access for Mac installations – as a fallback when help files aren't found locally.	
Added a text file to the `Mods\AdvCiv\Settings` folder explaining its purpose.		BUG and K-Mod have this folder, containing ini files of the Unit Naming and Custom Domestic Advisor components. BULL instead has a folder `Mods\BULL\UserSettings` that contains all ini files used by the mod. The presence of these files prevents BUG/BULL from creating them under `My Games` (cf. [this](#) post by EmperorFool).
Rationale	The folder is confusing in two ways:	

	a) The name suggests that it contains important global settings, but the settings are actually only for two particular mod components, which are even disabled by default. b) The contents of the folder are copied to `My Games\Beyond The Sword\AdvCiv\Settings` when the mod is started for the first time, so it seems that the folder in `Mods\AdvCiv` is redundant – which is true, but only once the copy has been created, so the Settings folder has to be included in the download archive.
Tbd.	If I'd store *all* ini files in `Mods\AdvCiv`, a copy under `My Games` would no longer be created. This would also make it easier to uninstall the mod. That said, it can lead to issues with Windows rights management ([example](#)), and fallback behaviors for that would be too difficult to implement for me. And I wouldn't want to include all the ini files in the download; this can perhaps be avoided by giving `Mods\AdvCiv` higher priority than `My Games` in `BugPath.findSettingsFile`. Either way, I should consider a different name for the `Settings` folder (`SETTINGS_FOLDER` in `BugPath.py`). Though, ideally, the game should then continue to check the old location so that players don't have to update manually (by moving their ini files).
See also	[K-Mod Git commit](#) that might force settings to be copied to `My Games` (haven't looked into it in detail).

`CvTextScreens.cpp` removed from `CvGameCoreDLL`.

Rationale	Apparently doesn't get compiled into the DLL, so it doesn't serve any purpose.
Credits	alberts2 (Caveman to Cosmos mod): [SourceForge revision](#)

Removed a few DDS art files from the `Art\BUG` folder that appear to be unused (the file names appear neither in the DLL nor in any of BUG's XML and Python file): `Arm.dds`, `Foot.dds`, `GreenFace.dds`, `RedFace.dds`, `YellowFace.dds`, `Star.dds`, `Trade.dds`. The total compressed size of these is only 7.3 KB.

009b	BUG initialization (only relevant for developers)	
AdvCiv		*BUG*
When a Python script file is modified while the game is running, Python scripts will sometimes, in part, fail to be reloaded. The likelihood (race condition) of such an error depends on which file was modified. As workaround, it usually suffices to save the file one or two more times.		Reloading fails almost always and leads to an unresponsive UI and nonfunctional alerts until the game is restarted. So, unlike in unmodded BtS, Python changes at runtime aren't really possible in BUG-based mods. [This](#) post by platyping confirms that this is an issue with BUG and not just K-Mod/AdvCiv. Some of the errors are caused by the Civ4lerts and Field-of-View slider components. But there are probably also problems with circular dependencies and/ or the proper order of reinitialization. I suspect that these problems were introduced late in the development of the BUG mod because developing such a complex Python mod without the possibility of changing scripts at runtime would've been very difficult.
Rationale	I haven't been able to fix this entirely.	
Tbd.	Perhaps it's mainly a problem with change ("dirty") handler functions referenced by `ChangeHandler.handle` (`BugOptions.py`)?	

009c	Removed Map Finder, BUFFY integrity checks	
AdvCiv		*K-Mod*
The configuration files, text files and Python scripts of the Map Finder mod component have been removed, and references to them commented out. One of the BUFFY game text files has been removed, the other mostly deleted, BUFFY code in Python kept commented out.		[Map Finder](#) is included in the BUG mod, so it seems that karadoc disabled it (by commenting out a few lines in `BugMapOptionsTab.py`.
Rationale	Generally, I don't want to remove any BUG/BULL features entirely, even if I don't think I'll ever use them personally, but Map Finder and the BUFFY checks are really features for competitive HoF players, which doesn't make any sense in a mod like this. I don't think it can be used as a developer tool either, e.g. to find out how commonly some very poor or powerful combinations of tiles occur, or only with major modifications. The compressed size of the removed files is 12 KB (negligible), but it also reduces the number of files to browse through when making changes to other BUG components.	
Config	Not terribly difficult to re-enable: Restore the removed files from BUFFY (K-Mod hadn't made any changes), uncomment any code marked with change id 009c.	
See also	The space on the BUG menu is now used for [004m](#), [004h](#) and [004z](#).	

009d	More graceful handling of bad data in BUG ini files	
Config	`BugOptions.py`, `BugOptionsTab.py`	
AdvCiv		*BUG*
When the BUG menu is opened and the index number for a dropdown menu read from an ini file (`My Games\Beyond The Sword\AdvCiv\Settings`) is not within the valid range, the default (set in `Assets\Config`) is used by the menu and stored in the ini file, replacing the invalid index number.		The index number is treated as 0 in such a case, meaning that the BUG menu (correctly) shows the first menu item as active. The invalid index number remains in the ini file. Index numbers in ini files can easily become invalid if the range of a `list` option is changed in `Assets\Config`.
Rationale	I'm changing some option ranges for v0.95. Using the 0th item is clearly inferior to using the default, and probably not what the BUG developers had intended either.	
Tbd.	My changes don't correct the invalid index until the BUG menu is opened. That's acceptable for now because I expect that players (if any) who have used the fairly exotic values that I've removed will take a look at the revised BUG menu after updating to v0.95. It should be possible to fix this in `getIndex` (`BugOptions.py`), but somehow I haven't been able to figure it out.	
Fixed some minor bugs in the error handling code in `BugOptionsTab.py`. These bugfixes are tagged with id 001.		
No comments are written to BUG ini files. If comments are already present (from a version of AdvCiv earlier than 0.95), those comments remain unchanged.		When the mod is launched for the first time (and no settings are stored from an earlier installation), it creates an ini file for every xml file in `Assets\Config` and fills the ini with the defaults

	set in xml. (This is still the case in AdvCiv.) Additionally, BUG adds a comment to each value in the ini file that includes, again, the default and the hover text for the option, read from the corresponding `...Options.xml` file in `Assets\ XML\Text`. These comments are, as far as I can tell, never updated, so if the default or hover text changes from one version of AdvCiv to the next, the comments won't reflect this unless the user clears the settings.
Rationale	Ideally, BUG would read the comments in the ini at startup (along with the actual values), check if they're outdated and update them if necessary. I don't think the comments are read at all though, so this would be quite an effort to implement. Updating the comments always isn't easy enough to do for me either, and might affect performance. I don't think the ini files should be manually edited anyway, so comments really don't seem necessary. They also take up a little bit of disk space.

010	Worker stealing disabled	
AdvCiv		*BtS*
Settlers and Workers die when attacked; no captured Worker appears.		Settlers and Workers die and the attacker receives a captured Worker. The AI immediately disbands captured units.
Rationale	Unlike human players, the AI doesn't know how to look for exposed Workers and steal them; nor how to disband Workers that are about to be stolen; nor how to guard Workers on border tiles. If the AI could do these things, it would be largely impossible to steal Workers from the AI, and players would have to guard their Workers on border tiles, which would be very tedious. This might be why the BtS developers (Blake?) abandoned efforts to let the AI steal Workers (`CvUnitAI::AI_poach` is never called). So, Worker stealing as a game mechanism can only work against an unwitting AI. I do like the slave-raid flavor, but I don't think it's worth putting the AI at a considerable disadvantage. Early disruptive wars remain useful for destroying AI Workers (see Tbd.) and for keeping enemy tiles from being improved – I wouldn't want to make this tactic unviable. Flocks of Workers captured in conquered cities also contribute to Worker under-employment, which tends to slow player turns down. Worker stealing also undermines difficulty settings that give the AI a free Worker. The Dynamic Diplomacy changes (decay of AI memory) would make Worker stealing even more attractive than it is in BtS.	
Config	Purely an XML change; can re-enable Worker stealing by restoring three values in `Civ4UnitInfos.xml` (look for `advc.010`).	
Tbd.	162 (rule changes against surprise wars) already makes it a bit harder to attack Workers and could be extended to make such attacks impossible. Additionally or alternatively, Workers should have limited charges; this would be another way to nerf Worker stealing. If attacks on Workers remain possible, AI improvements would be desirable. See	

	discussion on CFC here.
	See also the wip section (under id 010).

Show a message on the main interface when a noncombat unit is destroyed through an attack.	No message. Can't happen for workers, but can happen for ships in port and landed aircraft. For captured workers, there's a message.
Rationale	So that attacked workers don't disappear without notice. But also useful to have such a message for units that can't defend because of their domain type.

Increased the distance threshold that the AI uses when deciding whether to delete a captured unit.	
Rationale	Perhaps the AI should never delete captured units as in <u>Rise of Mankind</u>. In BtS, they're usually deleted.
	Only relevant if Worker stealing is re-enabled.

011	Decay of invested Worker turns	
AdvCiv		*BtS*
Once per round, when no progress has been made on any of a plot's unfinished Worker builds for the 8th turn in a row, the number of invested Worker turns is reduced by one for all unfinished builds in that plot.		No decay of invested Worker turns.
Rationale	BtS allows for some very fiddly micro-optimization, especially pre-chopping but also e.g. pre-building of Forts to protect strategic resources.	
	I don't want players to worry about leaving an improvement unfinished for some turns, e.g. until a Barbarian unit has been dealt with, but I want decay to be fast enough to make pre-chopping an irrelevant tactic. This balance is a bit difficult to get right.	
Config	Can be adjusted or disabled through `DELAY_UNTIL_BUILD_DECAY` in `GlobalDefines_advc.xml`	
011b	Partial builds	
AdvCiv		*K-Mod*
When the Ctrl key is held down while issuing a build command, the build is stopped one turn before completion.		BULL has an option to always stop short of completion when the build removes a Forest. K-Mod hasn't adopted this.
When Ctrl is held down while hovering over a tile, the progress on all unfinished builds is shown. Added an option on BUG's map tab to always show that information.		K-Mod shows no such info. BULL shows it if the "Partial Builds" option is enabled.
Credits	Some code adopted from BULL; that code is tagged with **011c**.	
See also	This open K-Mod issue states that the BULL implementation isn't safe for multiplayer games – which I can confirm. My own implementation is multiplayer-safe.	
Rationale	The pre-build command isn't really necessary now that build progress decays, but didn't take me long to implement, and it's something that other mods might want to adopt as my implementation is more flexible (can e.g. also be queued) and works in multiplayer. Also, players might disable the decay through XML.	
	Showing partial progress in game text is helpful in any case. Showing it through the Ctrl key seems like a good solution for everyone; the BUG option is mainly there to	

	make players aware of the feature. I've also written a loading screen hint for that purpose.
Tbd.	One downside of my implementation is that keyboard shortcuts (e.g. Alt+C for chopping) don't work when Ctrl is pressed. They do work in BULL when the "pre-chop" option is enabled, but then all chop commands have to be issued twice, which is, I think, a much bigger downside. I could make keyboard build commands work by using a letter key, say Y, instead of (or in addition to) Ctrl. Only the modifier keys Ctrl, Shift and Alt interfere with the build shortcuts; I suppose that's why karadoc decided to use the X key for suppression of unit cycling. However, pressing Alt+Y+C (or even Ctrl+Alt+C) to pre-chop really isn't convenient, so I don't think I'll bother with this.
	Like BULL; I'm showing a message when a build is suspended. That's perhaps more annoying than helpful. Also, the message would be timed better at the start of a turn. Currently, Workers abandon their build missions right after making the move that brings the progress to *x*-1 of *x* turns (perhaps this should happen at the start of the next turn instead) and the message is shown; often, Workers don't move until the player presses "end turn", and then the message is shown only very briefly, which looks strange.

012	Forest/Jungle defense reduced	
AdvCiv		*BtS*
Forest and Jungle provide no defense if the attacker owns the attacked tile, otherwise 25% defense. (Feature attack/defense bonuses from unit abilities or Woodsman promotion still apply regardless of tile ownership.)		50% regardless of ownership.
Rationale	Some players complain that removing Forests in the inner ring of a city is a no-brainer because the defense bonus for invading armies is too dangerous. I'm more bothered by the implausibly high defense bonus from Forest and Jungle, which leads to game-play problems too, such as Barbarians refusing to attack fortified units. Forested chokepoints are difficult to handle for the AI.	
	It makes some sense that units can defend well in forests (forests offer material for palisades etc.), and it makes sense that the civ that knows its way around in a forest (i.e. the tile owner) has an advantage; I'm assuming that these two factors cancel out when units in a Forest are attacked by the tile owner.	
Config	Tag `RivalDefense` added to `Civ4FeatureInfos.xml`. Can set that to 0 and `Defense` to 50 to restore BtS behavior.	
Tbd.	The Woodsman promotions should provide a (net) attack bonus against Forest and Jungle. Currently, only Woodsman III does.	

014	Capitulated vassals don't pursue victory strategies	
AdvCiv		*BtS*
Capitulated vassals can't be elected AP or UN leader, and can't be on the ticket for diplo victory.		A vassal votes for its master unless the vassal itself stands to be elected; no restriction on that.
Rationale	When even a capitulated vassal has more votes than the master civ's biggest rival, then the game is decided, and the master civ should win a diplo victory with the votes	

	of its minions. And of course a capitulated vassal shouldn't win.
AdvCiv	*BBAI/BtS*
Capitulated vassals don't pursue victory strategies, don't build team projects and don't build wonders of the world except Shrines.	Capitulated vassals pursue victory strategies and can actually win the game.
Rationale	Internet and Manhattan are things that the master may not want the vassal to build; SDI is covered by the master (change 143b); the other projects are for space victory. Capitulated vassal shouldn't build wonders that the master might want to build or that could hurt the master (UN, AP); it's simplest to block them all.
See also	130v about capitulated vassals voting along with their master and generally behaving like zombies. 112 about voluntary vassals breaking free when approaching victory. 143b scraps nukes upon capitulation.

015	Changes to Great People (GP)
Tbd.	Will probably allow Priest and Artist to hurry certain wonders; move culture bomb to Spy.
015a	Changes to tech flavor values
See also	020: Changes to non-tech flavor values. Tech flavor is special because it affects the GP discover abilities, while the other flavor values are mostly only relevant for the AI.
AdvCiv	*BtS*
No production flavor for Constitution. I.e. can't be discovered by a Great Engineer.	3 production flavor, 4 gold, 2 culture, 2 growth. Thus a tech that Great Engineers can (and may have to) discover.
Rationale	Engineers inventing constitutions is too far a stretch.
Tbd.	Fascism is also dubious (6 production flavor).

016	Extra tile yields from random events not added to city tile yields
AdvCiv	*BtS*
On city center tiles, extra yields from random events are added to the natural tile yield before raising the yields to at least 2 food, 1 production, 1 commerce.	City tile yield is computed as the yield from terrain, hill, unimproved bonus and river, all assuming that features (incl. Flood Plains) are removed. The result is raised to at least 2 food, 1 production, 1 commerce. Finally, extra yields from random events and yield bonuses from Golden Age are added.
Rationale	Should be treated like all the other yield effects; counterintuitive in BtS.
Credit	Pointed out here on CFC by traius.
Tbd.	Considering to change the city tile yield formula so that bonuses are applied after the raise step.
See also	004b shows the city tile yield before founding
Count extra yields even for impassable tiles.	No natural yields on impassable tiles. Non-natural yields (e.g. extra yields) can make

	impassable tiles workable.
Rationale	So that Python modders can make peaks workable.
Credit	[Post](#) by CFC user xyx
See also	[057](#): Changes to impassable terrain

017	AI trains fewer units when its military is already very large and drafts less in general
See also	[121b](#): AI hurries production less, especially units. 018: Reduced impact of Crush strategy [107](#): Fewer AI defenders [110](#): Changes to AI military buildup

AdvCiv		*BtS*
The probability of training a unit in a city is reduced based on the military power of the strongest potential enemy. Not as much when pursuing a military victory strategy.		The military power of other civs affects the number of trained units only indirectly through the "Area AI type"; generally keeps building units so long as the maintenance is affordable.
Rationale	Better to develop the economy more than to train excessive armies. AI stacks of doom can also get too disheartening if the human player is behind when Drafting and Rifling become available. Need to be careful not to throttle military production too early when aiming at a military victory. Those often have to race against a peaceful victory of a rival and may require overwhelming numbers to succeed in time. Moreover, the bigger an empire gets, the more units are tied down as garrisons (while still counting toward the military power rating).	
Decreased the base probability to train a military unit by 2 percentage points overall. Added an upper and lower bound for the city-specific train-unit probability based on the number of cities: The AI only gets to use very high or low probabilities once it has about five cities.		By the midgame, the probability is effectively 3 higher than set (per AI leader) in XML because the experience from Barracks is added in.
Rationale	The average probability to train a unit was about one in three, which seems a bit much in a situation where there is no war on the horizon.	

AdvCiv		*K-Mod*
AI unlikely to build Barracks before training a Settler for founding a second city.		About three times more likely than in AdvCiv.
Tbd.	Want to give Barracks a tech requirement.	
Don't draft away more than a third of a city's population unless defenders are urgently needed. (K-Mod has the same condition but doesn't apply it to Rifleman.) Don't draft at all when there is neither a war plan, nor a reason to reduce population.		Draft as many units as allowed except when there are angry citizens or upkeep gets too costly (that's a high ceiling). Stricter rules when drafting sth. other than Riflemen.
When it's not urgent, only draft with a per-turn probability equal to the normal `buildUnitProb`.		No randomness in drafting.

Reduced AI utility value for Nationhood civic when already paying high unit upkeep.	Utility is only reduced when `maxUnitSpending` is exceeded.
Rationale	The K-Mod behavior leads to huge AI stacks once/ if the AI hits Rifling and switches to Nationhood. If the player can stay out of harm's way through diplomacy, it becomes too easy to overtake the AI economically. One third of the population is still a lot. Drafting without planning war is generally a bad idea; can draft units pretty quickly once they're needed. Probably shouldn't adopt the Nationhood civic without war plans either. The change to civic evaluation may help with that. Don't want to make war plans a hard requirement for Nationhood though – that would give away the AI's intentions. The probability should make AI drafting less abrupt, and reduce it a bit overall.
Tbd.	`buildUnitProb` should be taken into account in a more meaningful way: currently mostly slows AI drafting down. However, need to change the drafting rules before putting effort into the AI; drafting is too efficient currently.
The target number of warships to escort a naval assault is adjusted based on the number of coastal cities owned by the enemy and the game era. Potential bug fixed in BBAI code that may have lead to large AI fleets when only a "minimal attack force" had been intended. The AI trains fewer sea explorers if the file name of the map script is "Pangaea".	Only the number of escorted transport ships matters, and whether they can defend themselves (Galleons).
Rationale	No need for an expensive escort if the enemy has very few ships. Can't be sure of that, but the number of coastal cities should be a pretty good predictor (and don't want to count enemy ships that the AI can't see).
Tbd.	Should check if the enemy can even train any dangerous ship; no need to protect (Industrial-era) Transports if the enemy ships are Frigates. Once that's implemented, should probably use `relativeNavyPower` if [UWAI] is enabled.
See also	[905a] buffs Trireme and stops the AI from using (and training) Caravels as escorts. Not sure if this has lead to fewer or more AI ships. [081] is supposed to train more ships when they're needed.
AdvCiv	*BBAI*
Reduced the number of attackers that the AI trains when there is no war plan.	BBAI added that behavior for "*building [a] city hunting stack*" that is also supposed to "*to produce early rushes on tight maps*" (comments in the code). May train as many as 12 attackers for this.

017b	Dynamic changes of unit AI type
AdvCiv	*BtS*
Caravels (and Carracks) trained by the AI as attackers (`UNITAI_ATTACK_SEA`) can turn into explorers (`UNITAI_EXPLORE_SEA`) if there is nothing to attack and there are too few	Land units can turn into (land) explorers, and there is a narrow rule that can turn a Galley into an explorer after transporting a Settler; but Caravels can only act as explorers if they've been

explorers.	trained for that purpose. Explorers can adopt various other roles if exploration isn't needed anymore.
Lowered the priority of guarding seafood when in `UNITAI_RESERVE_SEA`.	
Relaxed conditions for changing from `UNITAI_SETTLER_SEA` to `UNITAI_ASSAULT_SEA`.	
Galleys and Work Boats can change from `UNITAI_EXPLORE_SEA` to any other sensible type when Caravels are available.	AI won't train Caravels for exploration if it already has enough exploring Galleys.

Rationale	Should lead to fewer Caravels; that's why I'm grouping this with change 017.
Tbd.	There are probably other sensible AI type changes that the BtS code doesn't consider. A comment by the Vanilla developers also suggests this: "move some of this into a function? maybe useful elsewhere."
	However, one must be careful to keep UnitAI and CityAI/PlayerAI consistent, otherwise, the AI can end up training more and more unnecessary units.

Fixed an issue in BBAI code that had caused the AI to train at least one ship per sea area for transporting Settlers, even if those areas didn't have access to any city sites. Combined with the Unit AI type changes above, this had lead to large stacks of Galleys in water areas that didn't need ships at all.

018	Impact of Crush AI strategy reduced	
AdvCiv		*K-Mod*
The "Crush" strategy no longer causes the AI to train more units overall, and doesn't shift its yield focus to production. On the contrary, the AI trains slightly fewer units with Crush.		Crush makes the AI prioritize production. It also (no change in AdvCiv) lets the AI mobilize some of its defensive units as invaders, and trains more city attackers instead of defenders.
The AI doesn't adopt the Crush strategy until it's clear that the enemy doesn't pose much of a threat.		At least with Aggressive AI enabled, some AI leaders adopt Crush in any "total" war that isn't going badly.
		The Crush strategy has been introduced by BBAI.
Rationale	In K-Mod, Crush seems to be used in situations where the AI might be able to win a decisive victory by concentrating all its resources on the war. This leads the AI to go all-in more often than I'd like. The AI can easily end up training far more units than would be needed in these all-out wars, which sets it on a road toward a military victory. The AI goes for military victories too often in K-Mod.	
	Now the AI should use Crush only to bring war to a quick conclusion when it's winning anyway.	
See also	[115](#) and [104c](#) also make the AI less willing to go for a military victory.	

019	Lower impact of Aggressive AI mode (AAI) in BBAI/K-Mod code	
AdvCiv		*K-Mod*
Only minor impact of AAI in some of the BBAI and K-Mod code.		Especially K-Mod behavior depends on AAI in many places.
Rationale	For experienced players, I think there is a happy medium between aggressive and	

	non-aggressive AI, and really no need for two modes. Even for inexperienced players, it would be better to tie AI aggressiveness to the difficulty setting instead of a separate game option. (To be fair, BtS does that too: `iAIDeclareWarProb` is based on difficulty.)
	In the original Civ 4 code, the non-AAI behavior is very passive, especially on the low and medium difficulty settings; this has been a recurrent player complaint since the Civ 4 release. Therefore, I haven't reduced the impact of Aggressive AI in the original code.
See also	AAI can only be disabled through XML; see chapter on UWAI. 120b: No impact of AAI on malicious espionage.
	Posts by Blake about Aggressive AI in BtS. (Note, however, that the option existed already in Vanilla Civ 4.)

020	Changes to AI flavor values
See also	015a deals with tech flavor changes that determine which techs can be discovered by Great People.

AdvCiv	BtS/Warlords
Military flavor added to buildings that reduce maintenance; reduced on buildings that increase city defense. Gold flavor no longer associated with government center (reduced distance maintenance) and reduced on Courthouse. Gold flavor of corporate HQ reduced. Culture flavor added to modern entertainment wonders (Hollywood, Rock'n'Roll, Broadway) and Growth flavor reduced. Culture flavor also on Temples and Cathedrals (in addition to Religion), and on some misc. wonders. Growth flavor added to some happiness buildings (Colosseum, Market, Broadcast Tower) and National Epic. Added Religion flavor to Apostolic Palace (how was that missing?).	Since Warlords, each AI leader has one or two flavor values, and prioritizes buildings (and technologies) that match those values.
	Military flavor is on buildings that increase city defense, grant free XP, increase unit production or reduce war weariness. *Gold* flavor for extra trade routes, increased gold rate or reduced maintenance. *Culture* for some of the buildings that generate culture (Monument, Stonehenge, Theater, National Epic, Hermitage, Sistine Chapel, Eiffel Tower). *Growth* on buildings that improve health or spur population growth, and on a few that increase happiness (Notre Dame, Globe Theater). *Espionage* flavor for buildings that produce espionage points or Great Spy points. Religion for religious buildings. *Production* for buildings with (generic) production bonuses.
Plus some minor changes. I went through all buildings. Many buildings that had just one type of flavor now have two. In these cases, I've usually avoided assigning the full 10 points to any one flavor type (to avoid extreme preferences of AI leaders that match both flavor types).	
Espionage flavor removed from West Point, Pentagon and Forbidden Palace. Instead, AI governors consider flavor when evaluating Great Person points.	

Rationale	Military leaders don't usually want to get on the defensive and, therefore, shouldn't build Walls and Castles. I'm not removing the military flavor entirely because warlike leaders are also prone to counterattacks, and, (pre-)historically, militaristic cultures tended to build fortifications. Note that Protective leaders are still more inclined to build defensive buildings because of the production discount from the Protective trait; many of the Protective leaders have Military flavor.

	Reduced maintenance leads to a high science rate, which conflicts with buildings like Market that increase gold output. Gold flavor does give some incentives for growing wide (trade route bonuses), but I still think it's more important not to have the same flavor on Courthouse and Market. The next best choice seems Military; militaristic leaders tend to conquer wide empires.
	Corporations aren't there for generating gold. The HQs pair well with +gold buildings, but Gold-flavored leaders try to build Markets everywhere, not just in the HQ cities.
	The missing Culture flavor on Hollywood etc. seems like an oversight. While Cathedrals are more of a religious thing, they're so instrumental to Culture victory that they should also have Culture flavor.
	If Growth is supposed to support tall cities, it needs an even mix of health and happiness.
Tbd.	The Trade Mission ability of the Great Merchant also clashes with the plus-gold-rate buildings.

Apply the tech flavor values multiplicatively.	Additively; i.e. a flavor-based value between 0 and about 100 is added to the utility value of each tech. The utility value tends to increase with the economic output of a civ.
Reduced the impact of flavor on building evaluation, so that flavor typically makes about a 20-30% difference.	Additive in BtS, K-Mod replaced it with a multiplicative formula. Flavor can make up to a 100% difference, though 40-50% seems more typical. Since the building evaluation also affects the tech evaluation in K-Mod (not in BtS), building flavor indirectly affects tech evaluation too.
Rationale	The BtS formula means that the impact of flavor decreases with the economic output, so that a civ starts making more rational choices when it's doing well, and that flavor matters more when all eligible techs have a low utility value. None of this makes sense to me.
Tbd.	The flavor values in `Civ4TechInfos.xml` look loopy; need an overhaul. Doesn't take into account that the utility value counted for unlocked buildings includes the flavor values assigned in `Civ4BuildingInfos.xml`. Should perhaps treat units and civics in a similar way, i.e. assign flavor values through `Civ4UnitInfos.xml` (most would simply get sth. like 5 Military flavor) and `Civ4CivicInfos.xml` (no flavor tag yet) and count them only indirectly in the tech evaluation. This way, the flavor value would be reduced along with the unit value when a tech doesn't immediately unlock a unit, and the flavor values would not have to be changed when making changes to tech requirements.
	Should ignore first-to-discover abilities when assigning flavor values: already addressed through custom code.
	Revised flavor values will change GP "bulb" paths though; bad for players who have them memorized.

021	Adjusted map scripts with simulated tectonics; see also chapter [PerfectMongoose]().
See also	Since these map scripts frequently produce mountain chains and large areas of uniform terrain, the improved AI handling of areas separated by Peaks ([030]) and the reduced probability of large resource lumps ([129]) are especially relevant.

	Also tend to generate oblong continents near the poles, and 027 improves the selection of starting areas in such cases.	
021a	Tectonics	
AdvCiv		*BtS*
Uses the latest version 3.16 (Nov 2008) of the Tectonics map script. *"[...] added more rivers. [...] Terra option now has a nice looking Arabia instead of some landbridges and islands."* source		Uses version 3.15. (Although Dresden's Unofficial Patch had included the update to 3.16 and was, otherwise, mostly adopted by BtS 3.19.)
All map scripts are allowed to place Jungle on Plains. The impact on most map scripts seems very minor; most tiles at the equator are Grassland anyway. A sample (non-Tectonics) Pangaea map had only 8 Plains Jungles and 40 Grassland Jungles. (Not sure if that means that there would have been 8 fewer Jungle tiles without my change.) Banana, Sugar and Ivory can be placed on Plains Jungle (in addition to Grassland Jungle).		Jungle can only be placed on Grassland. Since Tectonics maps have lots of Plains near the equator, these maps end up with very little Jungle. The following resources can be placed on Jungle: Oil (Grassland), Uranium (any terrain), Banana (Grassland), Pig (Grassland), Rice (Grassland), Dye (Grassland), Gems (Grassland), Ivory (Grassland – or non-Jungle Plains), Spices (Grassland, Plains), Sugar (Grassland).
Credits	It's LDiCesare's map script. It was included in BtS patch 3.17, and he updated it once more after that.	
Config	The Jungle-on-Plains change is done in XML (`Civ4FeatureInfos.xml`).	
Rationale	I've not allowed Dye to be placed on Jungle Plains because the Jungle is almost impossible to see then. It's also a problem with Grassland Jungle Dye, but somehow it's not quite as bad – and players are used to it. Not important for gameplay; the map generator normally still finds enough Grassland Jungle to place the target number of Dye resources.	
Tbd.	Let Fractal-based map scripts place more Plains in the low latitudes. It's a better representation of the yellow soils in Southeast Asia than Grassland. This would also result in a nice even split between Grassland and Plains (currently the former is more common). Consider a rainforest feature/terrain for the equatorial belt. Allow Pig and Dye on Plains, perhaps instead of Grassland. Shifting resources away from the best terrain type (Grassland) should be healthy for game balance.	
See also	165 reduces the grid size of Tectonics.	
AdvCiv		*Tectonics 3.16*
Tectonics landmass type options "Earthlike" 80%, 70% and 60% water. Reduced the number of rivers for all landmass types, and a bit more for Pangaea.		Only 70% and 60% Was supposed to be reduced (perhaps to the level of v3.15) only for Pangaea, but due to an apparent bug affected (only) 60% water instead.
Rationale	Corresponding to High, Medium and Low sea level. My 80% option actually does the same as the 70% option in the original script, the new 70% corresponds to the old 60%, and the new 60% is a new setting. I've noticed that far less land is generated than the old percentages said. My percentages are still too high. Actually, the ratios vary a lot. The new 60% option sometimes does lead to 40% land, but at other times just 23% ...	

	3.16 generally had more rivers than e.g. Fractal; I didn't like that. Not sure if Pangaea really needs special treatment. Seems to receive fewer rivers in any case (albeit longer ones than on maps with smaller continents).
Tbd.	Make the script produce reasonable land ratios reliably. Should be the same as Fractal; see the frequencies stated under 021b.
Credits	4Dingo4 and LDiCesare have suggested formulas for the 50% setting here.
021b	PerfectMongoose (PM)
Credits	I've adopted a few changes from Toffer90's `C2C_World.py` script (Caveman2Cosmos). Apart from that, see chapter PerfectMongoose for credits.
Config	The "Perlin Noise" option is equivalent to the "PerfectWorld 3 Landmasses" option in PM 3.3. The "Plate Tectonics" option is equivalent to "Perfect World 2 Landmasses". This option affects continent sizes and shapes and the distribution of hills and peaks. Perlin noise has more varied results, landmasses tend to be smaller and snakier and hills and peaks more widely distributed.

The "Break Pangaea" option has been removed from the menu; it's now implied by "Old World Start".

All options that I've removed should still be fully functional; they just can't be selected on the menu anymore. (I.e. it's easy to restore them by editing the map script.)

To make PM maps reproducible, in addition to setting fixed RNG seeds in `CivilizationIV.ini`, the Python RNG needs to be disabled (`self.UsePythonRandom = False`) in PM. It should not be necessary to disable the Python RNG in multiplayer games: out-of-synch errors reported about the predecessor PW2 have been fixed by AIAndy.

Some debug output gets written to `PythonDbg.log` if Python logging is enabled. Unfortunately, this can't currently be toggled in the script. |
| *See also* | The latest version of my changes that should be fully portable to other mods and unmodified BtS: Git commit
After that, I made changes that are specific to AdvCiv. That said, I've also made some more non-AdvCiv tweaks that could (easily) be merged into the portable version. Maybe I'll do that at some point and post the result in the PM thread on CFC. This version I had posted earlier is now outdated.

I've used version 3.3 as my starting point. LunarMongoose never made that version available as a standalone script, so I took it from the latest version of MongooseMod, version 4.2a. It's the same version that cephalo has posted here in the PM thread. Then I've stripped away all the code specific to MongooseMod. (Realism Invictus has taken the same approach – its version of PM is also based on v3.3.)

In Oct 2018, cephalo published "PerfectWorld6" as a Lua script for Civ 6 (link), which "*might be the best PerfectWorld yet.*" The Perlin noise generator doesn't seem to have been altered (same as in LunarMongoose's Civ 5 port), apart from a small tweak to the "twist frequency" that I had already arrived at independently. The rainfall formulas are also unchanged. Most of the new code is dedicated to lakes and rivers: "*I have […] incorporated lakes into the river system, so that rivers can flow into lakes, and lakes can flow into other lakes […].*" To me, this doesn't seem worth the trouble of attempting a back-port.

The LoR SDK ModMod makes some changes to PerfectWorld2, but these are probably obsoleted by LunarMongoose's work. Changes by Fuyu in RevDCM are minor and obsoleted by my own changes. Likewise those by Antmanbrooks for Realism Invictus. Mongoose Mod makes numerous changes beyond those in the standalone version of the script (see in particular the v4.1 release notes), but nothing |

	that I'd like to merge (or very little).
Tbd.	My changes are getting extensive enough to justify a name change. I'd like to lose the nonsensical "Mongoose" part. "Mundus," in a way, means both world and perfection and would still be a nod to LunarMongoose's user name. Or "Ad Mundum" – that would not cover the perfection part (not what I strive for anyway …) but allude to the AdvCiv mod; and it sounds like a motto for modders ("to the world").
See also	Unless the starting position iteration algorithm ("SPI"; 027) is disabled, the PerfectMongoose algorithm for starting positions only designates the New World when the Old World Start option is used.
	I've used the mapstat log for measuring the relative frequencies of terrain, feature types, elevations, rivers and resources on Fractal maps (customized by change 129). I've tried to get PM to match those frequencies. For reference, these are the relevant Fractal statistics under default settings: Total tile count: 4368 (84x52); Land: 20.4%; Resources per player: 21.63 Land breakdown: Hills: 16.7%; Peak: 5.5%; Grassland: 50%; Plains: 27%; Desert: 12%; Tundra: 4.5%; Snow: 4%; Jungle: 14%; Oasis: 0.2%; Flood Plains: 1.5%; Forest: 21.5%; River plots: 20% Water breakdown: Ice: 20%

AdvCiv	PM 3.3
Same map sizes as Fractal. I've also aimed at the same terrain frequencies (see above) with some exceptions: I've set the land-sea ratio, and frequency of hills, forests and plains a little bit higher – but still far lower than PM 3.3. My Tundra frequency is (on average) closer to PM 3.3 than to Fractal – around 10% – at the expense of Grassland, which lands somewhere between 40% and 50% on average. Toward the poles, land becomes less likely to occur (59% "attenuation"). Eliminated the elevation artifacts resulting from attenuation.	A larger tile grid is used for all map sizes, leading to about 40% more tiles. The land-sea ratio is about 28%. On the other hand, Grassland and Plains cover only 40% of the land area and much of the Grassland is covered by Jungle. Hills are almost twice as common as on Fractal maps, making arable land even more scarce. No attenuation at the poles (though PerfectWorld6 does use 75% attenuation). The attenuation factor (if used) also affects altitudes above 0; altitude differences caused by attenuation can make peaks and hills more common near the poles.

Rationale	Apart from using different overall terrain frequencies, PM generates, locally, more uniform terrain than Fractal, i.e. larger deserts, steppes and mountain ranges (clusters of peaks and hills). I see that as a strength as it gives the various regions of the map more character and a more Earth-like feel, and results in city sites of marginal value that can present a third choice between stopping to expand and waging war. As cephalo's pitch for PerfectWorld6 puts it: "*a believable map that makes exploration more fun and adds extra challenge to the game.*" The balance problem posed by civs starting near poor terrain should be much lessened by AdvCiv's starting position algorithm.
	Keeping the Tundra frequency relatively high is in line with these considerations. It's really a matter of the amount of land generated in high latitudes; Fractal (and also e.g. Pangaea) generate very little land anywhere near the poles. PM could easily mimic that through a very low attenuation factor – but shouldn't in my opinion. That being said, without attenuation, it's pretty common that large continents get connected by some sort of elongated Antarctica. This is unrealistic (armies can't traverse a polar desert) and doesn't play well either as it diminishes the importance of ships.
	It's important to me that the same player counts can be used for PM as for the standard map scripts. To this end, I've set a slightly higher land-sea ratio for PM than for Fractal because city sites on PM maps tend to be spaced a bit farther apart and

there tend to be more small, initially uninhabited continents that civs can't immediately expand onto. On the other hand, PM maps have longer coastlines than Fractal maps and therefore more seafood and a higher number of viable coastal city sites.

It would be nice if the land-sea ratio were closer to the real-word value, which should be around 28% if we assume that Antarctica is mostly represented by the ice sheets shown above and below the map edges in Globe view. One could change this for PM or all map scripts and keep the player counts as before by decreasing map sizes. Maybe this would place continents too close together, especially when using PM's Old World Start option. There's always the Low sea level option. Generally, the maps don't have to be scale models of the Earth, but getting somewhat close to the terrestrial proportions would be nice.

See also	Some discussion about the terrain proportions: CFC link

Tweaked the method for placing peaks and hills so that the altitude of both the lowest orthogonal and lowest diagonal neighbor is taken into account and so that water tiles are treated as having higher altitude than the seafloor; removed the Absolute Height option.	Peaks and hills are placed based on altitude differences unless the "Absolue Height" option (introduced in v3.3) is enabled. cephalo's scripts use the difference between a tile's altitude (as generated by either plate tectonics or Perlin noise) and the mean of its neighbors' altitudes. LunarMongoose uses the minimum of the neighbors instead of the mean. All these methods result in peaks and hills forming bigger clusters than on Fractal maps. The difference-based methods – and also absolute heights when combined with plate tectonics – result in a bias for peaks and hills on or adjacent to coastal tiles.
Rationale	Hills really mustn't clump together too much; some terrain here and there with highly unbalanced yields can be fun, but not large swaths of it. Clumps of peaks don't have this problem, but similar methods need to be used for peaks and hill placement, otherwise peaks won't have surrounding foothills. Placing hills based on absolute altitude is pretty obviously bad for gameplay, and hills also aren't a convincing representation of plateaus,which may well be irrigable; – so I've removed that option. Apart from finding a good balance between representing Himalayan-size mountain ranges and spreading hills out, there is also the issue of arranging peaks, hills and flat tiles into natural-looking patterns. Distinguishing orthogonal and diagonal neighbors seems helpful in that regard.
See also	[030](030) makes coastal peaks less likely for all map scripts that use the standard terrain generator. See the rationales there against peaks on coastal tiles.
Tbd.	I think hills still form bigger clusters than I'd like, especially with the Plate Tectonics option. I don't think the slightly higher overall frequency of hills (compared with Fractal) fully compensates for the uneven distribution. Straight, uninterrupted chains of peak also still occur too frequently. Perhaps one should simply check for such formations explicitly and break them up probabilistically. Would also be nice to check for tiles completely enclosed by peaks and either open up an entrance or turn the enclosed tile(s) into peaks as well.
Hard latitude limits for Jungle. That means, on maps with little tropical land, Jungle can only get denser to compensate (to an extent) and sparser when there is much tropical land; generally, the occurrence of Jungle varies a lot depending on the latitudes of the continents.	A fraction of the hottest and wettest tiles become Jungle, meaning that the latitudes where Jungle can occur expand when there is little land near the equator and shrink when there is much land near the equator. In v3.2, the temperature requirement was still absolute, meaning that the Jungle coverage was more dependent on the

	latitudes of the continents.
Rationale	A fixed frequency of Jungle tiles isn't important for game balance. Too much Jungle can be a problem; that can be addressed well enough by making the Jungle sparser. Jungle in higher latitudes looks very jarring (and I don't think players will interpret it as temperate rainforest).
Especially wet plots are represented as Plains rather than Grassland. Jungle can occur on plains but is a bit more common on grassland.	All wet plots become Grassland. Still, more plains occur in the tropics than on other maps (that use the default terrain generator). Jungle is placed only on grassland. As a result, jungles are less dense than on other maps.
Rationale	Plains are a slightly less unrealistic representation of tropical soils than grassland (at least in terms of food yield; the color of plains also fits for clay-rich soils). Also, allowing jungle on both plains and grassland makes the jungles less scattered.
Avoid placing Jungle adjacent to Desert.	In theory, the rainfall map should be somewhat smooth, which would avoid Jungle next to Desert. Perhaps it's less of a problem on larger maps.
Rationale	I like the alpine tundra idea, but snow-covered mountains (or jungles) next to desert are jarring (even on small maps, where such contrasts may not be so unrealistic considering the scale of the model).
Tbd.	Desert next to Grassland is still quite common.
Snow and Tundra only appear in the polar latitudes with the exception of Snow hills, which may occur on high-altitude tiles with at least moderate rainfall in temperate latitudes.	Tundra and Snow are placed on any tiles that are sufficiently cold and (in the case of Tundra) wet. Tundra and snow are used (also) to represent alpine tundra, i.e. in plots that have a high elevation but aren't rugged or high enough to make them impassable.
See also	004g renames the "Ice" terrain type to "Snow".
Credits	Adopted some code from `C2C_World.py` that adjusts per-tile temperature values to the tile's distance from the sea.
Rationale	Mixing some Snow hills into clusters of Peaks seems like a nice way to represent mountain passes. That should be intuitive enough; I don't think players will interpret it as a polar desert randomly existing at medium altitude in a temperate zone. Apart from this special case, Tundra and Snow at medium or low latitudes look too strange, especially near the desert or jungle belts.
Tbd.	Maybe Tundra hill would be a better choice for mountain passes; representing also alpine tundra. I don't think both terrain types should be used – this only concerns a handful of tiles on most maps.
Oases don't have to be surrounded by desert; a few adjacent non-desert tiles are OK. No cold deserts near the poles.	Oases only on tiles entirely surrounded by desert. Sometimes large deserts are placed right next to tundra.
Rationale	Oases that are totally surrounded by desert are rarely useful.
Attempt to turn lakes fed by a river into bays only	Comment in the script (by Cephalo probably):

with a 50% probability.	"*It looks bad to have a lake, fed by a river, sitting right next to the coast. This function tries to minimize that occurrence […].*"
Rationale	I think those lakes are fine if they don't occur too frequently; don't want to "minimize" their occurrence.

The PM river generator gets to place some of the rivers, the rest are placed by the standard river generator in the DLL. And I've tweaked the PM generator to discourage rivers in extreme latitudes.	By default, v3.3 lets the DLL place all rivers. Optionally, the PM river generator (pretty much unchanged since PerfecWorld 2) can be used instead. The PM generator is based on the rainfall map and tends to place most of the rivers in extreme latitudes, especially near the equator. This can leave few rivers for the temperate areas, and Desert Flood Plains are also pretty rare. The PM rivers also tend to be shorter than the rivers placed by the DLL.

Rationale	River placement should take into account how crucial a river is for human habitation. Rivers through the rainforest have, historically, not been as important as, say, the Nile or the Huang He. Covering the tropics with rivers also exacerbates the problem of ahistorically high tile yields in that region.
	I don't want to detach river placement entirely from the precipitation model, and some more short rivers (placed by PM) make coastal regions more valuable, which increases realism and may also, indirectly, help the game balance by making naval units more relevant. Letting the DLL place some of the rivers seems like a good compromise that should ensure a reasonable coverage with rivers across the map.
Tbd.	Toffer90 has made extensive changes to the river generator in `C2C_World.py`, but his code is based on a lake generator that doesn't look like it's straightforward to port.

The balancing ("normalization") of the tiles near starting plots is handled entirely by the DLL functions that most other map scripts use for normalization.	In v3.2, all the standard normalization functions are disabled and replaced with custom code that seems very reluctant to make any changes: No rivers or lakes are added, no bad terrain (tundra) converted, jungles and peaks only removed when they occur in large numbers. v3.3 has re-enabled the addition of rivers and lakes through the DLL.
Rationale	The PM normalization code can lead to awful starting locations. BtS might do a bit too much balancing (especially for a map that's supposed to look natural), but, fortunately, I've already made changes (id 108, 027) that make the balancing less aggressive.

Added some code that encourages one starting site per continent to be placed inland. However, this only matters if the Starting Position Iteration (SPI) algorithm is disabled through XML; when SPI is enabled, the DLL takes over the assignment of starting sites.	The PM code for assigning starting sites tries to (after narrowing the search to a few dozen candidate sites) maximize the distances between starting sites. This means civs frequently start on the tip of some peninsula (where they may easily get boxed in) and virtually never on a non-coastal tile.
Rationale	The PM code also suffers from using custom functions ("potential value") for tile evaluation, which, I'm confident, are inferior to the ones in the DLL that have been improved a lot through changes 031 and 027.
Tbd.	The potential value functions are still used for splitting up the map inot New and Old World for the Old World Start option. Ideally, the found value function in the DLL should be used instead – or the DLL should handle the split entirely (Old World Start

	game option usable for all map scripts).
	It might also be nice to use the starting sites assigned by PM as the initial solution of SPI. That's awkward to implement; see comment in PM's `findStartingArea` function. Or, if the current initial solution (i.e. the BtS algorithm in the DLL) actually works better, then PM should not waste time computing its own starting position. Seems a bit difficult, though, to separate the Old/ New World split from the computation of specific starting sites.
Resources are placed in the standard way (`CvMapGenerator`). Fur can only be placed on flat tiles, and only above a latitude of 35°. This applies to all map scripts.	Custom code for resource placement. Seems to be largely the same as in `CvMapGenerator`. I guess cephalo wanted to make some changes without changing files other than the map script. The BtS map generator places Fur on flat and hilly Tundra and Snow tiles at any latitude. Since PM uses Tundra and Snow (usually as hills but not always) to represent high elevations, Fur can appear fairly close to the equator. Deer is restricted to latitudes above 30°.
Rationale	I want my own, more extensive, changes ([129]) in `CvMapGenerator` to apply instead of the PM changes. (The PM `BonusPlacer` class also seems to be quite slow, but that's only based on one sloppy test.) The Fur change isn't really relevant for PM anymore now that I've restricted Tundra and Snow based on latitude. It could matter for other map scripts. Fur near the equator could represent savanna fauna, but, in BtS, that's clearly not the idea. 35° is still pretty far away from the poles, but, in combination with the no-hill restriction, it should be OK. Fur on hills doesn't make perfect sense to me anyway as the most widely known fur-bearing animals (beaver, ermine, mink) live near water, which tends to be more abundant in flatlands (mountain valleys too, I guess). Perhaps fresh water should be required?
Option for using the PerfectWorld 2 Climate System removed. Also can't opt for the hex-based PerfectWorld 3 landmass generator. I've kept the Perfect World 2 landmass generator as an option (renamed to "Plate Tectonics").	Options "PW3 Generator (Square Grid - Accurate)", "PW3 Generator (Hex Grid - Interesting)" and "PW2 Generator" for landmass generation; and "PW3 Climate System" and "PW2 Climate System" for climate.
Rationale	I want to remove experimental options in order to make the script easier to use. The hex-based generator seems like an intermediate result of back-porting the Civ 5 code. I guess it adds some noise to the process; doesn't look particularly interesting to me. As for the Climate Systems, it looks like cephalo ported the PW2 system to Civ 5, made some adjustments, and then LunarMongoose copied the Civ 5 code back to PM, keeping the older version as an option.
Tbd.	Should probably just delete the PW2 climate code (so far, I've only removed the option from the menu).
"Break Pangaea" map option removed. Enabled if and only if the Old World Start option is set. Reduced the maximal number of meteors and their diameter. Inside that diameter, the elevation is decreased. The decrement is maximal in the center and becomes gradually smaller toward the edge of the crater. This way, there are usually no Peaks on coastlines created by meteors.	If the Break Pangaea option is enabled (which it is by default), the script checks in the end if there is a single group of continents (spearated only by coastal water) containing more than 70% of the land tiles; if so, circular groups of land tiles are removed (explained as meteor strikes) until all continents have at most 70% of the (remaining) land tiles. The impact sites are chosen based on

After each meteor, the sea level now gets adjusted to restore the land-sea ratio. (Meaning that a meteor will eliminate land tiles in one place and -indirectly- create new land tiles in other places.) The meteors can strike only land.	(I think) the betweenness centrality metric. The number of meteors is limited to 15, the average diameter appears to be about 9 tiles depending on map size. All tiles in the diameter have their elevation set to 0. Since Peaks are placed (in a later step) based on differences in elevation, coastlines created by meteors are (always?) covered by Peaks. Since the total number of Peaks to be placed is limited by a target ratio, few or no Peaks get placed inland. The meteors may strike coastal water. This doesn't necessarily turn the coastal water into ocean because that's a matter of distance to land and not elevation.
Credits	The sea level update is inspired by `C2C_World.py`. CFC posts by Toffer90 on the subject: 1 | 2
Rationale	"Meteor" works as a metaphor for explaining the Pangaea split algorithm, but the meteor impacts shouldn't be easily indentifiable on the map – meteors don't really alter coastlines, and the impact sites could also leak map knowledge.
It's not guaranteed that the meteors succeed in creating a New World that is significantly larger than the islands that PM tends to create regardless of the Break Pangaea option. At least at Medium and High sea level, the success rate seems pretty high, let's say closer to 100% than to 50%.	It seems that the meteors usually succeed, but they may remove so much land that the map looks like a plucked goose and the Old World becomes very crowded.
Rationale	Relaxing the limit on the number of meteors (i.e. moving back toward the PM limit) could increase the success rate. Since the land-sea ratio gets restored, this wouldn't render the map unplayable; I only worry that it'll look too peculiar.
See also	I've added a recommendation about the number of players to the label of the Old World Start option. This is consistent with change 137, which adds such recommendation labels for Low and High sea level.
Latitude is taken into account when identifying a suitable continent (or group of continents with a shallow-water connection) to serve as the New World.	Only based on tile counts; so the New World or Old World may have far fewer habitable tiles than anticipated.
Rationale	Latitude is a simplistic heuristic; a quick stopgap change.
Modified the Perlin noise and (more so) Plate Tectonics landmass generator to encourage larger continents.	Both map generators rarely, if ever, produce a continent of Eurasian proportions. The Perlin noise (PerfectWorld 3) often yields results similar to the "Snaky Continents" option of other map scripts. For PerfectWorld6, cephalo has made the same tweak as I (increased `twistMinFreq`) to the Perlin noise generator.
Rationale	More massive continents are more Earth-like and make diplomacy more interesting. Should occur at least sometimes.
Tbd.	Still not massive enough. I don't think the Perlin noise generator has a suitable parameter. Reducing the resolution (`hmWidth`, `hmHeight`) should, in theory(?), result

	in coarser structures, but lowering the resolution quickly results in block artifacts. I guess one would have to combine Perlin noise with a different input – e.g. the Plate Tectonics (PerfectWorld 2) generator.
When placing lakes, the map areas are now recomputed only once in the end. I'm not sure if this is purely a nonfunctional change, but at least I'm not seeing any adverse effects.	They're recomputed after every added water tile. This adds a couple of seconds to the total map generation time.
Tbd.	Could probably improve the performance substantially by moving some code into the DLL. That said, it's difficult to determine which parts are slow. So far, I've only stepped through the DLL-to-map-script calls in the debugger to get an impression. The lake computation stuck out a bit but was still only a small portion of the total time.
The elevation map (i.e. early in the process of generating terrain) gets shifted sideways by an offset in a way that minimizes elevation values at the western and eastern map boundaries.	Both landmass generators seem to make some effort to avoid land at the map boundaries, but the result is not reliably optimal.
Credits	Based on code in Totestra (Git commit). Though the latest, "streamlined" version has thrown that code out (Git commit).
Rationale	Still doesn't reliably avoid land at the boundaries. It's not even possible to do when continents overlap horizontally. And my algorithm optimizes a (weighted) sum of elevation values; a more exact result could be achieved by calculating the sea level threshold and counting the plots above the threshold. However, making use of the target land-sea ratio this early in the map generation process might be a bit messy. In any case, based on some limited testing, the current solution does improve things.

022	Changes to AI paranoia	
See also	107 also deals with the AI's defensive strategies 109 sets Economy Focus when no threatening civ is known 130u deals with attitude values assumed by the AI for human players UWAI also makes use of paranoia ratings for its "third-party intervention" aspect of the war utility calculation.	
AdvCiv		*K-Mod/ BBAI*
The paranoia value of an AI civ is based on the attitude and personality of rival civs, not vice versa. Human rivals are assumed to be moderately bellicose with an adjustment based on the AI civ's attitude toward them. Made a similar change in the computation of `CityThreat` (non-immediate military threat toward a city); using a mix of defender and attacker attitude there. Decreased impact of rival victory strategy. Increased impact of geographical closeness; except when not on the same landmass. Some other, minor changes.		Paranoia is computed as a function of vicinity, power ratio, our attitude and personality, their victory strategies and whether we're their worst enemy. Paranoia determines if the BBAI strategies "Economy Focus" (low paranoia) or "Alert1" and "Alert2" (high paranoia) are adopted (neither if medium paranoia). Paranoia is greatly increased if a rival is at stage 3 of Domination or Conquest victory.
Rationale	A K-Mod comment in the CityThreat function says, "*For good strategy, this should*	

	probably be their *attitude rather than ours. But perhaps for role-play it is better the way it is.*"
	For `CityThreat`, I think both arguments have merit, so I'm averaging the attitudes in this case. As for paranoia, producing additional defensive units against a benevolent rival makes little sense roleplay-wise, and can be quite damaging for the AI, therefore, only use the neighbor's attitude in that case.
See also	[107](#) changes the computation of closeness between civs.
Reduced paranoia if the threatening civ is so powerful that resistance is likely futile.	The more powerful they are, the more paranoid we get.
Rationale	"Things without all remedy should be without regard."
Increased paranoia based on the threatening civ's tech era; up to a factor of 1.5 in the Future era.	Paranoia ratings are not adjusted to the game progress.
Rationale	AI civs become more willing to conquer faraway cities over the course of a game (because maintenance becomes less of an issue and unit mobility increases).
Tbd.	Perhaps the "closeness" values that enter into the paranoia calculation should already be adjusted to the game progress.

023	Occupation countdown based on revolt probability
AdvCiv	*BtS/ K-Mod*
The occupation timer (after conquest or a revolt) is decremented only with a per-turn probability equal to the tenth power of 1 minus the revolt probability.	The occupation timer is decreased by 1 each turn.
When a city is conquered, the occupation timer is set to the minimum of 3 and the population size.	Occupation after conquest last for 3 turns plus 50% of the population size, e.g. 13 turns in a size-20 city.
A revolt sets the timer to 2 plus the number of prior revolts.	Also 2 in BtS and in K-Mod 1.45 (but 3 in earlier K-Mod versions).
Damaged units have their culture garrison strength reduced proportionally to the damage.	Unlike combat strength, garrison strength is unaffected by damage.
The per-turn chance to decrease the occupation timer is shown on the main interface (city tile help text) and on the city screen (Nationality bar help text).	

Rationale	BtS occupation times get too long in Renaissance; one reason why it's difficult to catch up at that point. 10 turns is a very long time when the game may last just 100 more turns, and the city may still have to spend some 20 turns on essential buildings before it starts paying off.
	By basing the occupation duration on the revolt probability, I hope to reward players that conquer only one or a few cities at a time and can then afford to sit in them to make occupation end quickly. Generally, occupation durations are still going to increase as the game progresses – mature cities tend to have high revolt probabilities. Now, 3 turns is often just the time that units need for healing anyway, so this would hardly slow down conquests if it weren't for the reduced garrison strength of damaged units.
	I'm not showing a message when an occupation counter is decremented; could become too much when occupying several conquered cities.
Config	The conversion of revolt probability into the probability of decrementing occupation can be tweaked in `GlobalDefines_advc.xml`. Can also restore the BtS rules there (longer but deterministic occupation).
See also	101 reverts the K-Mod changes to revolt probability. 099c makes revolts more common by allowing them to happen outside of foreign culture range. 023 wouldn't really work without 099c because, in BtS, conquered cities often have 0% revolt chance. 210b displays an alert when occupation ends.

Revolts can't happen in occupied cities (conquest or prior revolt) if the city owner is at war with the cultural owner. If they're not at war, a revolt in occupation is possible but mitigated by the following special rules: • The revolt test is only executed if the decrement-timer test has failed. • Revolt probability is halved while in occupation • A revolt during occupation does not increase the occupation timer; it does increase the revolt counter and can flip the city. Being at war does not increase culture garrison strength.	No revolts during occupation. Culture garrison strength doubled while at war.

Rationale	If revolts were impossible during occupation, a player could withdraw all units to deliberately prolong occupation, which makes the city worthless for the current owner – but also for the cultural owner (can't flip).
	Regular revolts during occupation would be too punishing though. The halved probability only makes up for the loss of garrison strength when units are damaged as part of a revolt.
	I don't like having complicated special rules for revolt during occupation, but I don't see a better solution. Or perhaps a city should flip deterministically after spending 10 consecutive turns under occupation (unless at war)? That would be a bit simpler.
	It's a bit problematic that cities can remain under occupation indefinitely while at war, but the above rules would lead to small probabilities of potentially painful revolts pretty regularly. I guess a grace period would have to be added (there's already a timer for cultural ownership after conquest), and maybe if foreign culture strength were halved (quartered? martial law does make it easier to put down a rebellion) instead of halving revolt probability, it could work, but that wouldn't work so well for peacetime occupation – would usually drop revolt chance to 0 once a revolt occurs. (Or maybe that's OK?) Pretty sure that allowing revolts while at war would further complicate the rules. Revolts while at war would also have to be counted as a war success of the revolt (AI) player.
See also	Brief [CFC discussion](#) about revolts under occupation.

An occupied city heals units only as fast as a friendly non-city tile (15 HP per turn).	An occupied city heals as fast as an unoccupied one (20 HP per turn) except that Hospital doesn't count while in occupation.

Rationale	For plausibility and to slow wars of conquest down a bit more. 10 HP per turn (heal rate in neutral territory) might be even more appropriate, but then players could heal units faster by moving them onto some owned tile adjacent to an occupied city, which would be counterintuitive and tedious.

024	Order in which AI contacts other AI randomized	
AdvCiv		*BtS*
When contacting other AI civs for trades, the AI goes through them in a randomized order.		Fixed order based on the player slot id.
Rationale	Some deals aren't mutually exclusive, but the AI can e.g. give away a resource only once, and may only have enough gold to convince one civ of a joint war. The fixed order leads to a slight bias for trading with AI civs with low ids.	

025	Reduced culture spread by capitulated vassal onto master's tiles
AdvCiv	*K-Mod*
The tile culture spread by a city onto a tile is halved if the city owner is a capitulated vassal of the tile owner.	Capitulated vassals spread their culture normally.
Rationale	Even if cities can't flip to vassals (change 099c), the revolt-inciting culture spread is still a good reason not to accept capitulations. It's OK if capitulation isn't always the correct choice, and perhaps this is an interesting dilemma, but capitulation should be correct more often than not, and 099c makes it much harder overall to keep revolts in check.
Config	Through `GlobalDefines_advc.xml`
See also	130v neuters capitulated vassals in several ways

026	More gold offered in AI-proposed deals
See also	134a gives humans a discount when suing for peace.
AdvCiv	*K-Mod*
When an AI civ contacts a human with a trade offer, when trying to balance both sides of the deal, the AI is willing to give away more gold than normally. If the player negotiates, the extra gold is no longer offered.	The AI can offer deals that are slightly more favorable to the player, but the portion of the AI treasury offered for trade isn't increased.
Rationale	The aim of the K-Mod change was to make it "worth considering the deal the AI offers [...] rather than going straight to the renegotiate button" (from the K-Mod 1.07 changelog). But a worthwhile offer still seemed very rare, so, while worth considering, it didn't exactly hurt to never consider AI proposals.
Config	`AI_OFFER_EXTRA_GOLD_PERCENT` in `GlobalDefines-advc.xml`
Tbd.	A general problem with any one-time offers: If the player can't just click "renegotiate", then the player doesn't know what else the AI might have. Encourages the player to routinely enter the Foreign Advisor during diplo. The only fix I can think of (other than removing the one-time offers) is to have the AI remember its initial offer. Then, if the deal is renegotiated, that offer remains available through "what would make this deal work": If the player offers to give the AI what it had originally asked for (or that and sth. in addition), or asks for what the AI had originally offered (or a subset of it), the AI suggests the original offer unless it finds an even better one. This way, the player would again always renegotiate, but would still take a look at the original offer in order to possibly get back to it.
	For now, I've added a loading screen hint about opening the Advisors during diplo.
	Another way to make one-time offers more attractive: Increase the trade value counted for "fair and forthright" trade if an AI offer is accepted without renegotiation (but don't increase the trade value for trade with worst enemy). Or even decrease fair-trade memory if an AI offer is rejected. Or, another idea: Relax trade denial checks, e.g. by randomly treating the relations modifier as a couple of points higher when making attitude-based checks while putting together an AI-to-human trade offer.
	Another issue: AI-proposed tech trades are often worse than the trade resulting from "care to renegotiate" → "what would make this deal work?". Should check `AI_counterPropose` before offering a tech trade to the player.

027	Changes to the selection of starting sites; starting position iteration (SPI) algorithm
See also	[021b](): PerfectMongoose has its own algorithm for assigning starting locations, which SPI, if enabled, supersedes. [108](): "Normalization" of starting sites (based on the results of SPI). [108b]() may swap starting sites between players (based on the results of SPI). [031](): Changes to AI city site evaluation – which is the basis for the initial selection of starting sites. Many of those changes also factor into the evaluation of the space for expansion performed by SPI. [CFC thread]() about SPI

AdvCiv	*BtS*
If the map script doesn't override any starting position function, then SPI treats the position found by `CvGame::assignStartingPlots` as an initial solution (to the problem of finding a balanced starting position, i.e. an assignment of one starting site to every civ) that it tries to improve on iteratively. Upfront, SPI computes a selection of alternative starting sites, typically 5 to 10 times as many as the number of civs in the game. Those sites are selected in a way that balances maximal found-city values (evaluating only city radii, not surroundings) against maximal dispersal across the map. Each iteration then considers moving one or two of the current starting sites to alternative sites. If a (single or double) move is found that significantly improves the attributes of the position, then SPI commits to that move. Then the next iteration starts unless the new solution is already deemed good enough or a time limit is reached. SPI also terminates if none of the considered moves would be an improvement. I'll briefly describe below how starting positions are evaluated, i.e. what attributes are treated as desirable, and which moves are taken into consideration. If a map script overrides `findStartingArea`, then SPI considers only alternative sites on landmasses where at least one starting site is located in the initial solution. I've modified PerfectMongoose so that it overrides `findStartingArea` instead of `assignStartingPlots` when the Old World Start option is used and otherwise leaves the starting position entirely to the DLL. SPI will freely change starting positions in non-team games on Pangaea. For Pangaea team games and for any other scripts that override `assignStartingPlots` or `findStartingPlot`,	There are three DLL functions concerned with starting positions that map scripts can override: `assignStartingPlots`, `findStartingPlot` and `findStartingArea`. The following official/ bundled scripts override none of them: Fractal, Continents, Balanced, Big_and_Small, Medium_and_Small, Hemispheres, Great_Plains, Tilted_Axis, Ice_Age. Likewise, K-Mod's not_too_Big_or_Small doesn't override any starting position functions, nor does AdvCiv's Mixed_Continents ([advc.mxc]()). Those scripts leave it to `CvGame::assignStartingPlots` in the DLL to assign a starting site to each player. That function greedily assigns a starting site to one player at a time by calling `CvPlayer::findStartingPlot`, which, in turn, calls `CvPlayer::findStartingArea` to select the least crowded landmass, and then selects the starting site with the highest found-city value on that landmass. The found-city value computation penalizes short air distances to any starting sites already occupied by other players (`CvPlayer::startingPlotDistanceFactor`), and takes into account workable tiles on the same landmass in a 11x11 (K-Mod: 13x13) square centered at the prospective starting site. The scripts Lakes and FantasyRealm override only `findStartingArea`. I think in a pretty inconsequential way since those scripts create only one major landmass anyway; so they basically work as described above. The remaining scripts (offical, bundled and also all scripts from the PerfectWorld family) override `assignStartingPlots` or `findStartingPlot`. Most of them merely limit the set of potentially valid starting sites and then let the global `findStartingPlot` function defined in `CvMapGeneratorUtil.py` greedily select the

SPI takes no action.	valid sites with the highest found values. Effectively the same thing as `CvPlayer::findStartingPlot` in the DLL except that the crowdedness of the landmasses isn't explicitly taken into account.
	Donut and Pangaea only allow coastal starts; the latter applies that restriction only in team games.
	Highlands, Boreal and Rainforest don't restrict starting positions but perform some cleanup around the sites found by `CvMapGeneratorUtil.py`.
	Tectonics, Earth2, Terra and NewWorld rule out New World starts. Earth2 also has more than 15000 tiles at Huge size. GlobalHighlands has nearly as many; also problematic.
	The other scripts impose various non-trivial restrictions.
SPI also handles players in scenarios that have no fixed starting coordinates, i.e. players with `RandomStartLocation=true` or any players that didn't get one of the plots designated as `StartingPlot`. And I've dialed down the randomness resulting from the `RandomStartLocation` flag.	Scenarios without fixed starting coordinates (`StartingX=`/ `Y=`) and without preplaced cities (`CityPopulation=`) use `CvGame::assignStartingPlots`. If those scenarios have plots flagged as `StartingPlot`, then those plots are randomly distributed among the players. If there are more players than starting plots, then `CvGame::assignStartingPlots` will find additional starting sites – as it would e.g. for Fractal. Exception: Players set to `RandomStartLocation=true` in the scenario file can't receive one of the designated starting plots; the WB scenario parser will instead call `CvPlayer::findStartingPlot` with `bRandomize=true`. That parameter adds (a lot of) random noise to found-city values. The BtS version of the Europe scenario is the only scenario that uses `RandomStartLocation=true`. The six starting plots from the Vanilla version of that scenario still exist in BtS, but are no longer used (because all players receive random starts).
Config	SPI can be disabled through `ENABLE_STARTING_POSITION_ITERATION` in `GlobalDefines_advc.xml`. For PerfectMongoose, the map's own algorithm can be re-enabled (to take precedence over SPI) in `PerfectMongoose.py` (search for "advc.027"). The exception for the Pangaea map script is hardcoded in `StartingPositionIteration.cpp`.
	Debug output can be enabled in StartingPositionIteration.cpp (`SPI_LOG`, `DEBUG_SPACE_BREAKDOWN`). For debugging, `MapRandSeed` and `SyncRandSeed` should be set to a value greater than 1 in `CivilizationIV.ini`; otherwise generated maps aren't reproducible. For PerfectMongoose, `self.UsePythonRandom = False` has to be set in addition in `PerfectMongoose.py`.
	Regarding scenarios with `RandomStartLocation`, the randomization of found-city values can make sense to avoid getting the same starting sites in every game. That

	said, the Europe scenario has random resources anyway, and, in any case, the randomness was overdosed.
Rationale	Desirable properties of a starting position (some obviously conflicting):

- Fairness – giving every civ a realistic chance to compete with every other civ. Some degree of unfairness is in my opinion also desirable because this magnifies the effect of the `StartingLocPercent` handicap (see 108b) and thus can be a fairly organic way to make the game more challenging.
- Tension – if sites are close together, the early game tends to be too tense, even if the respective civs all have enough room for expansion to, in theory, coexist peacefully. If sites are far apart, in particular, if one civ is alone its landmass, then there is too little tension in the first half of the game.
- Plausibility – the starting sites represent the cradles of civilization; they shouldn't be mediocre sites.
- Variation – a great variety of starting sites in terms of available yields, resources, freshwater, terrain features, coast/ inland, latitudes, distance to other civs and room for expansion increases replayability.

I think the BtS algorithm – with a fair number of tweaks by various modders – does a pretty good job at finding the next best site. The main problem is that it's a greedy algorithm, placing one player after another. The last site that gets placed and its neighborhood often have too little space. On concave landmasse, the use of air distances is another significant shortcoming. The PerfectMongoose (PM) algorithm is less greedy and uses path distances across land, but neither BtS nor PM can deal with shallow-water connections, which is a significant problem for PM because its maps tend to be quite watery. The PM algorithm also tries too hard to maximize distances between starting sites, which results in coastal starts all around.

On the bottom line, the unfair starting positions on "scraggly" maps like PM or the Big/Small family seriously affect their playability, and also Fractal maps that are supposed to be somewhat crowded (which is what the AdvCiv standard settings aim at) often end up giving one or several players far too little room.

The motivation for using an iterative algorithm is that an evaluation of a given starting position, while a complex task, is conceptually straightforward and computationally feasible, while I'd have no clue how to devise an efficient algorithm that somehow places all players simultaneously at their final starting sites. For centroidal Voronoi tesselations, iterative algorithms that move one centroid at a time (Voronoi iteration) have been used with some success. A Voronoi diagram isn't exactly what I need because, in Civ, a) not every map tile is equally important, b) the distance metric needs to account for obstacles, and c) tiles that are close to multiple starting sites should not be counted for just one site. Nevertheless, the two problems have similarities and the same type of algorithm might work for both. (That was my intuition anyway.) It's also nice that the BtS algorithm doesn't get discarded; it's put to good use for a strong initial solution.

Partitioning space is computationally hard. The pre-selection of alternative starting sites goes a long way toward making the effort manageable. Typically, only a small portion of land tiles are suitable as starting sites anyway (sufficiently high found-city value), and dispersal, i.e. the elimination of all but one or two potential sites in a cluster, only means that SPI isn't able to adjust starting sites by just one or two tiles; such fine-tuning shouldn't be necessary either. Still, among dozens or, on (super-)Huge maps, even a few hundred of alternative sites, it's infeasible to evaluate every possible starting position. So the iterative algorithm is still needed for a heuristic search in the solution space. I suppose that, given a function for evaluating starting positions, one could apply genetic programming or reinforcement learning, but a starting position evaluation is not cheap computationally and I don't think those

	generic, off-the-shelf algorithms are frugal enough. Also, devising my own algorithm was probably less work than integrating an external library. One further desirable property – that I had taken for granted initially: • Meaningful choice to move the starting settler Initially, SPI made it optimal to settle in place most of the time. The first decision in the game – arguably shouldn't be as consequential as it is – but even more importantly shouldn't be boring. I've taken a number of small measures to improve the situation: Increased the overall resource density a bit again (see 129); randomized the selection of potential sites a little bit so that they're not always quite optimal locally; increased found-city values a bit when there is a resource or other high-yield tile just out of reach; increased the lake placement chance (normalization; see 108) a bit again; introduced a bias for the inner ring when placing extra resources (normalization; see 108).
Tbd.	Would like to use SPI for all map scripts, but will have look at them one by one to figure out how SPI can respect the restrictions that they impose. E.g. for the New World scripts, it should be possible to have them override only `findStartingArea`. For others, it might be easiest to let the DLL ignore the restrictions in the script (going by the map script name as I've done it for Pangaea) and to reimplement those restrictions in the DLL. This way, the mod won't have to include modified copies of unconventional scripts that I don't want to be listed near the top on the Custom game screen. 030 treats land cordoned off by peaks as a separate continent. Does that solve the problem with having to open up "pockets" of peaks on Highlands, Boreal and Rainforest? Depends on how many there are; mustn't take too much land off-limits (as SPI would probably do) when assigning starting sites. Should be OK at least for Boreal and Rainforest; ought to have fewer peaks than Highlands. I've already adjusted the starting site evaluation (under id 027, but regardless of whether SPI is enabled) so as to avoid starts near a land area boundary. This is problematic on any map because, if the normalization step ends up breaking the boundary, a starting site that was previously isolated by a mountain chain can become unexpectedly powerful.

Some more specifics about SPI (rationales included):

When evaluating a postion, several statistics are computed that correspond to the goals of high fairness, high plausibility and medium tension stated in the "rationale" box above. As for fairness, a "start value" is computed for each of the current start sites, expressing how favorable the site is for whichever civ that starts there, and the distribution of those start values (especially its outliers) determines how fair the position is. The sum of the start values is a measure of the position's overall plausibility. Tension is covered by a "volatility value." All those values combine into an overall "starting position value."

The start value of a site is computed from its found-city value (counting only the city radius), a "space value" for the surrounding land, or really any land that a civ starting at the site might be able to claim, and modifiers accounting for possible trade and warfare. Those modifiers also take into account game options, most importantly "No Tech Trading" and "Always Peace." The "space value" is based on path distances and tile "yield" values (perhaps a bit of a misnomer because trade values of resources also factor in). All path distances and yield values are precomputed before the first iteration; doing so repeatedly during the iterations would not be computationally feasible. Since we're only interested in distances that involve potential city sites, the distance table takes up a few megabytes of memory at worst. I use Dijkstra's algorithm with a distance metric that assigns a high – but not necessarily prohibitive – cost to transitions from land to shallow water and from shallow water to deep water.

The path distances still don't fully solve the problem of civs getting cut off by a rival city at a chokepoint. Specifically, path distances fail when such a chokepoint is closer to the rival while

much of the terrain beyond the chokepoint is about equally close to both civs. I think this could only be addressed by actually simulating how the civs will expand from a given starting position.

The order in which moves are considered is important because the algorithm commits to the first decent move that it finds. Apart from saving time (by cutting the current iteration step short), this isn't necessarily a worse strategy than selecting the move that offers the greatest immediate improvement; such a move might lead into a local optimum. The algorithm considers moving outliers (in terms of start values) first; negative outliers have especially high priority. Sites with high volatility are also prioritized. Within the same continent, only moves to neighboring sites are considered, the rationale/ intuition being that the initial solution can often be made to work with minor local changes. The algorithm also considers destinations on other continents, but only a single site per continent; this leaves it to subsequent iterations to optimize the placement within that continent.

Moving just one site at a time doesn't seem promising, considering that every iteration step has to improve the start position value. Often, moving one site from *A* to *B* will give some other site(s) near *A* too much space or some site(s) near *B* too little, and the starting position value might only improve if other sites are moved to compensate and then further sites to compensate for those moves (ripple effect). The more the merrier, but I don't think moving more than two sites at a time is computationally feasible, and it doesn't seem to be necessary either as the current algorithm works well enough.

In team games, SPI computes starting sites as normal (see above) and then swaps sites between players in order to let teammates start near each other and to avoid one team getting outnumbered by another on any continent. A greedy algorithm is used. After SPI is through, the BtS algorithm may make further swaps, but only when swapping significantly decreases distances between teammates and never between continents.

Exception: For very large team games (high player count, high average team size), SPI isn't used at all.

See also	108b makes some minor changes to the BtS algorithm. I've summarized the BtS algorithm for starting site assignment in team games under that change id.
	CFC post by me
Rationale	The BtS algorithm doesn't take into account civ counts per continent. It's obviously very unfair when e.g. all three members of team A start on the larger of two continents together with a single of team B, while the remaining members of team B start on the smaller continent. In such a case, a symmetrical starting position (two members of each team on the larger continent) should be preferred. Fairness shouldn't always overrule vicinity though; players of team games are used to starting near each other most of the time, I guess it's important for the collaborative feeling. Also, having teammates start far apart reveals more information about the map early on, which diminishes the element of exploration.
	I could've modified the BtS algorithm so that the distribution of players across continents is taken into account, but I wanted to make use of the path distances computed by SPI (which take into account shallow-water paths), and that was easier to implement within the SPI class. The BtS approach of processing all possible swaps (of starting sites between pairs of players) in an arbitrary order is also a bit suspect to me. I've been too lazy to implement a sensible heuristic for ordering the swaps. My greedy algorithm doesn't work great though; therefore I'm still running the BtS algorithm afterwards with some restrictions. It seems to work especially badly for large (or rather Huge) maps with large teams (more than 2 members). Teams can get split up across three continents then even though the (main) continents are so large that one team outnumbering another isn't a major concern. For now, I'm leaving such maps completely to the BtS algorithm (i.e. no SPI at all). Perhaps there would be no harm in letting SPI at least make the initial selection of starting sites (but on Huge

	maps, SPI doesn't usually fare much better than BtS because of the time limit on the number of iterations) ...

In summary, the Starting Position Iteration algorithm is, so far, used for the following map scripts:

Fractal, Pangaea (except team games), Continents, PerfectMongoose, Mixed Continents, not too Big or Small, Big and Small, Medium and Small, Hemispheres, Balanced, Tilted Axis, Great Plains, Lakes, Ice Age, Fantasy Realm, Savemap (New Starts option).

It also fills in missing starting sites in the following scenarios when the player count exceeds the number of starting sites designated by the scenario:

Africa, Battle, East Asia, Eastern United States, Planet, South America, Europe
(In the BtS version of Europe, all sites get assigned by SPI.)

SPI is never used for maps with more than 12000 tiles. Huge normally has about 10000; Huge PerfectMongoose and Terra are a bit larger, but still under 12000. SPI also isn't used for large team games (specifically: if the product of player count and average team size exceeds 36).

Disable `MinStartingDistanceModifier` of Hemispheres and not_too_Big_or_Small.	16 of the official and bundled map scripts set a `MinStartingDistanceModifier` that affects the minimal distance at which the DLL can place starting sites: Hemispheres, Big_and_Small, Arboria, Boreal, Donut, Earth2, Global Highlands, Highlands, Rainforest, RandomScriptMap, Team_Battleground, Terra, Custom_Continents, FantasyRealm, Great_Plains, Mirror.
Rationale	Map scripts shouldn't needlessly interfere with the starting site selection. For the more eccentric scripts, it's probably fair enough. Custom_Continents: Don't want to copy that into AdvCiv.

AdvCiv	*BtS*
When computing the starting area score for a continent, count only those rivers and coastal tiles that aren't peaks, tundra, snow or dry desert. Take the land tile count times 0.5. Add 1.5 times the number of bonus resources. If SPI is enabled, then this change only matters for the initial solution.	When starting locations are not determined by the map script (and most map scripts don't), then a starting continent is chosen for each player by `CvPlayer::findStartingArea`. That function computes a score for each continent and selects the one with the highest score. Part of the score formula is the number of river edges, the number of coastal land tiles (to account for the water surrounding the continent) and the total land tile count. Bonus resources don't count directly, but the total of all tile yields does.
Rationale	Too many civs had been placed on continents near the poles.
Tbd.	Would be better to evaluate the surrounding water directly (than to count coastal land tiles); should be pretty easy to do through the Shelf class that I've added in change 300.
See also	kekm.35: Further changes to address the same problem. 031 disables K-Mod city evaluation code that was trying to steer starting locations toward an even distribution of players among landmasses.
If cheats are enabled, Shift+Ctrl+mouseover on a land tile without any units shows a breakdown	Various combinations of Ctrl, Shift and Alt reveal various internals for development purposes.

of the starting area score.	
Rationale	I've used this to figure out where the assignment of starting areas goes wrong, and left it in the code for future adjustments.
Any Tribal Villages located directly at a starting site get removed. Exception: Not in scenarios.	Tiles with tribal villages are disregarded when looking for a starting site.
Rationale	Finding good starting sites is more important than keeping the number and spacial distribution of Tribal Villages intact. (Tribal Villages that were deliberately placed in a scenario could be more important yet.)
When (re-)assigning starting sites to players, if the sites were selected by SPI, then the start values of the SPI final solution are used to determine which site is the strongest. If the site that would be assigned to a human player purely based on difficulty has a high volatility value (as computed by SPI) and a similarly strong site (in terms of SPI start value) has a significantly lower volatility, then the human player is given the low-volatility site.	Starting sites are selected without regard to which player will start at which site. Technically, the sites still need to be assigned to players; this preliminary assignment is arbitrary. The proper assignment is done based on `StartingLocPercent` from the difficulty level, which, apart from some scenarios, is going to be the same for all AI civs. I.e. human civs are supposed to receive a better or worse start than AI civs depending on the difficulty level. This is determined through found-city values computed with the `bStartingLoc` parameter (`bNormalize` in AdvCiv) so that tiles surrounding the city radius ("greater range") are taken into account.
Rationale	Found-city values aren't reliable when it comes to the space available for expansion (see somewhere above); they also put too much emphasis on the tiles within the city radius. SPI tries to avoid selecting high-volatility sites, but this isn't always possible. Not assigning those sites to human players should significantly reduce the ratio of unplayable starts encountered by human players. That's worth compromising on `StartingLocPercent` (although I like handicapping humans through their starting site). High-volatility can just mean isolated on a somewhat small landmass or quite close to rivals (but not necessarily boxed in). I wouldn't want to make such starts exceedingly rare for humans, so it's good that the volatility avoidance mechanism doesn't guarantee low volatility.
Tbd.	It would be better to not to (re-)assign sites until after the normalization step. As it is normalization can strengthen sites that looked weak, in some cases even drastically, in particular, when a removed peak removes a barrier between two continents.
See also	108b: Other changes to the reassignment procedure.
When deciding whether a starting site needs to be improved during the normalization step, the game makes use of data computed by SPI (if SPI was used for computing the starting sites). Both start values and found-city values (accounting only for the city radius) are used. This mostly applies to random extra resources and features, but can also lower the target amount of food for sites where food isn't already abundant.	Based on found-city values with the `bStartingLoc` parameter (AdvCiv: `bNormalize` – if SPI isn't used), i.e. taking into account the city radius and its surroundings. The food target is not affected by found-city values.
Rationale	Use the best data available.
See also	108: Other changes to normalization

027b	Store RNG seeds used for map generation
AdvCiv	*BtS*
The seed set for the sync RNG and map RNG at game start are stored in savegames and displayed in Debug mode when holding Ctrl while hovering over the big flag button on the main interface. When the map is regenerated, the stored map RNG seed is updated.	Maps are generated based on the map RNG, but the sync RNG determines the civ and leader selection at game start if "Random" is chosen during game setup. The two RNGs are seeded by the EXE(?) with some random number (perhaps from the system RNG) unless `MapRandSeed` or `SyncRandSeed` are set to a value above 0 in `CivilizationIV.ini`. The game doesn't store the seeds, so map generation is only reproducible if the seeds come from the `.ini` file.
Rationale	For development purposes. Reading the seeds off the flag button and entering them into the `.ini` file is tedious, but at least there is now *some* way to reproduce the map generation process when something unusual happens.
During the starting site normalization process, only the map RNG is used.	Normalization uses both RNGs, pretty haphazardly.
Rationale	To reproduce a regenerated map, the sync RNG seed of the original map needs to be used because only that seed will result in the same selection of civs and leaders. That seed mustn't be used for normalizing the regenerated map though. It's also just cleaner to commit to one of the two RNGs.

028	Submarines as escorts
AdvCiv	*BtS*
When a stack with invisible and visible units is attacked, the attacking player sees combat odds against the best visible defender, but an invisible unit may replace that defender once the attack is ordered, i.e. if the invisible unit is the better defender (considering unit cost and combat odds). Invisible units don't defend if all the team's units in the tile are invisible (same in BtS), i.e. an attacker can't stumble upon an invisible stack, and invisible units don't defend units of other teams.	Invisible units only defend when revealed (e.g. by a Destroyer).
Rationale	Not plausible for a Submarine to sit by while e.g. a fully loaded Transport gets attacked by another Sub. The new behavior makes Subs worse in situations when the defenders are outnumbered by modern ships, e.g. a Transport (or just a Work Boat) and a Sub against four Battleships. That said, players can easily prevent this by not stacking Subs with visible units. All in all, the change should make Subs (and Stealth Destroyers) a bit more useful.
See also	Depends on changes to `CvPlot::getBestDefender` made for [061](#).

029	Changes to air recon missions	
AdvCiv		*BtS/ BBAI*
Tiles revealed by a recon mission remain visible until the *end* of the subsequent turn of the recon unit's owner, or until the unit carries out another mission.		Visibility ends on the *start* of the owner's subsequent turn. If the options for showing foreign moves are enabled, the player still sees all foreign moves before the fog of war returns.
Consequently, it doesn't matter much if a recon mission is carried out early in a turn (manually), or after all other units have moved (auto recon or queued mission).		BBAI introduced auto-recon for air units. The changelog says: "*Air units can now be set to explore, they use the same explore logic as AI planes and then have extra logic if that doesn't push a mission. Note that planes on auto explore always move at the very beginning of your turn.*"
		The latter part doesn't seem to work; auto-recon (and queued recon missions) are barely usable because tiles are only visible from the end of one turn to the start of the next.
Rationale	During the owner's turn, nothing happens in the fog of war, so there isn't much of a point in hiding the tiles at the start of a turn. (It can make a difference for air strikes.) The advantage of my change is that players no longer need to remember which units had been visible, and it makes it easier to automate recon.	
Tbd.	Would still be nice the have a "Recon until canceled" mission that repeats recon on the same tile every turn (akin to the Blockade mission; on that note "Blockade for one turn" would be nice to have too). OrionVeteran had started to implement something like this but didn't finish it.	
	I'd like to nerf recon a bit by allowing units on recon to be intercepted, ideally even when not at war (and no OB). And reduce the number of revealed tiles (-2 radius), especially for non-recon air missions (-3 radius).	
AdvCiv		*BBAI*
Tweaks to the AI code that selects the recon target when automated: Consider the full recon range (currently 5x5 tiles) around each candidate target tile. Don't prefer water tiles. Prefer non-friendly tiles and especially enemy tiles.		Only tiles adjacent to a candidate target are taken into account. Unowned tiles and water tiles are preferred.
Rationale	It seems that the BBAI code was written with the (only) goal of spotting enemy transports. I want to be able to also spot incoming forces on land.	
See also	A crucial bugfix in one of BBAI's auto-recon functions (`CvUnitAI:: AI_exploreAirPlotValue`, `AI_exploreAirRange`) is tagged with advc.001.	
	650 also lets the AI use the auto-recon code.	
`iAirRange=-1` in Civ4UnitInfos.xml means infinite range. Can also be used for air strikes with infinite range.		
Rationale	For mod-mods; Dawn of Civilization uses this to represent satellites.	
Credits	from Dawn of Civilization (Git commit)	

030	Peaks can separate areas
Config	`PASSABLE_AREAS` flag in `GlobalDefines_advc`.

AdvCiv		*BtS*
When one or several peaks make parts of a landmass unreachable by land, that secluded part is treated as separate land area.		Only water can separate land areas.
Rationale	Should make things easier for the AI. The AI still has to be able to handle dynamic obstacles because of closed borders, and it can (but not that well).	
	Would prefer to base the rules for colonies and single-continent wonders on BtS landmasses, but that seems like too much work. When e.g. Notre Dame is completed in one area, would have to call `changeBuildingHappiness` on all areas with the same `RepresentativeArea` id.	
See also	Without advc.enum, the additional areas would probably increase memory usage (and thus cache performance) enough to affect performance and savegame size negatively.	
When shelf ice makes part of a water area unreachable, that secluded part is treated as a separate water area. Exception: In scenarios that let the civs start with more than just a Settler and Warrior (i.e. don't call `CvGame::setInitial Items`), water areas are as in BtS.		Only land can separate water areas. BBAI had tried and failed to implement separate areas for ice; see comments in `::areaValid` (CvGameCoreUtils).
(No change to the trade rules.)		Trade cannot pass through ice (nor peaks).
Had to revise a lot of code related to unit movement so that the AI recognizes that submarines can pass between adjacent water areas.		The AI handles submarines well but builds all sorts of ships in cities cordoned off by ice. Barbarian ships can also spawn trapped in ice.
The amphibious `canMoveAllTerrain` unit ability is not supported by my code. (Shouldn't crash, but the AI won't be aware of potential attacks.)		I don't think the BtS AI supports it either. Anyway, no unit actually has this ability. (Gunships can't enter water.)
When ice is added or removed (WorldBuilder, global warming), the water areas are not recalculated, i.e. the AI remains unaware if e.g. the Northwest Passage opens up.		
A city adjacent only to an ice-locked water area of fewer than 10 tiles can't construct buildings like Lighthouse that require an adjacent water area of at least 10 tiles.		
Rationale	Shipbuilding in ice-locked cities makes the AI look very bad.	
Tbd.	Can't recalculate the water areas in a running game because various statistics and AI data are stored at the `CvArea` objects; hence the exceptions for scenarios and global warming. Another issue is that placing a Peak through WorldBuilder currently leads to recalculated land areas, which should be fine when creating a scenario, but bad when play is supposed to continue after exiting WorldBuilder. Should bite the bullet and write code that preserves per-area data after recalculation. Starting site normalization can also remove peaks; not sure if those recalcs are safe.	
	Also, plot groups (for trade network) aren't currently updated after removing ice. (That's not an issue introduced by change 030, but still ...)	

	Or, if I can't do the above, I should at least change global warming so that Ice can only melt when it is orthogonally surrounded by water (so that water areas are unaffected by global warming).
	Could limit the (step) diameter of water areas (by limiting the search depth of the `calculateAreas_visit` function), which might help the AI when the geography resembles the American continent (minus the Panama Canal). Then again, distances between coastal cities are rarely long enough to make naval movement between them impractical.
	Regarding Lighthouses in ice-locked cities: Easier to implement this way. Also plausible that a lighthouse isn't helpful in a city that is locked away from maritime trade. Even more true for Harbor, Customs House, Great Lighthouse ...
	I've cut some corners in the AI danger checks, meaning that the AI won't recognize that an enemy submarine can attack across Ice that fully separates two water areas. Seeing that submarines are normally invisible anyway, this will probably never be an issue (and if 315 allows Explorers to enter Peaks, they'll still only be able to attack Barbarians, and the Peak will probably block the Barbarians' sight anyway). If there is a problem after all: I've left the code for handling such situations commented out in the AI danger functions; the performance penalty is still going to be very small.
See also	003b: Lists functions introduced for dealing with submarine movement across Ice. 033: The checks for naval blockade had relied on water areas not being adjacent to each other, so I had to change these checks. 051 is responsible for calculating areas in scenarios. 041 does allow Dry Dock and ship production in ice-locked cities.

Hover text on water tiles whose water area is adjacent to Ice and has a size of less than 10 says that the tile is "ice-locked".

Rationale	To give some warning to the player that cities won't be able to construct most coastal buildings (and that ships, while allowed, may not be able to move anywhere).
	While the new text is pretty low-key and players aren't going to inspect icelocked water tiles often, I still would've preferred showing "ice-locked" only when a settler is selected or when a city has been founded adjacent to the water area. But I think the latter part would be confusing because water areas without a city would appear not to be ice-locked.
Credits	CFC user crullerdonut pointed the problem out to me.
Tbd.	Show it only when a settler is selected?

AdvCiv		*BtS*
Fractal-based map scripts are two times less likely to place a Peak on tiles orthogonally adjacent to water than on other tiles.		It seems that Peaks are just as likely near water as inland. A Peak that is orthogonally adjacent to water can make part of a landmass inaccessible via land. (Inland Peaks could do that too in theory but would have to form a circle.) Almost(?) all map scripts bundled with Civ 4/Warlords/BtS are based on Fractal, but PerfectMongoose isn't.
Rationale	On Earth, a drop in height from above the tree line to sea level is rarely so abrupt that it would justify a Peak next to a water tile, though there are some examples, in particular in the Central Andes. Still, there is (always?) a coastal strip wide enough to be traversed even by an army, so Peaks completely denying land access to an area is not realistic. Then again, the Darién Gap does prevent traffic by land between the Americas. This is a case of swamps more than mountains, but Civ doesn't have a swamp terrain type. As a compromise, I'm making Peaks that block coastal movement	

	less likely rather than impossible.
	This discussion on CFC is somewhat related.
Config	Implemented in Python (`CvMapGeneratorUtil.py`)

030b	AI doesn't train cargo units for naval assault in land/ice-locked waterbodies	
AdvCiv		*BBAI*
Build cargo units for naval assault only in coastal cities that share a water area with an enemy city.		Build cargo units for naval assault in all coastal cities that don't have a land path to an enemy city.
Rationale	Change 030 generally prevents cities that aren't adjacent to "relevant" water areas from building ships, but doesn't address the specific case of building up for a naval assault.	
Tbd.	The tracking of city counts per water area may not be totally reliable in scenarios. See comment in `CvPlot::processArea`.	

031	Changes to AI found value: Revised most of the code, rearranged it a bit and made several additions	
See also	108 makes changes to the found value of the initial city (which is important for starting site normalization). 040 assigns a found value to unrevealed tiles. 052: changes AI found behavior in scenarios 007 makes some changes to the found values computed when the Alt key is held down in Debug mode. 036b: changes to resource evaluation (few so far; affecting both city placement and resource trade)	
Tbd.	I don't think the found value computation takes city specialization into account at all. The computation of the culture modifier is quite primitive. The culture rates of nearby cities should matter. There's also a K-Mod comment about that in (what's now) `AIFoundValue::calculateCultureModifier`. Perhaps introduce a personality-based factor that e.g. makes Willem (Netherlands) want to found cities more closely together (i.e. worry less about overlap) and Genghis Khan (Mongolia) farther apart.	
AdvCiv		*K-Mod*
AI considers settling on tiles without any bonus resource in the city radius if the city can expect to work at least 3 freshwater tiles. Also considers tiles without any "good" resource if there are at least two (not-so-good) resources.		Tiles without a good bonus resource in the city radius are disregarded; resources that the AI doesn't have access to yet and food resources are considered as "good".
Reduced the impact of distance from capital based on era. No distance penalty for small-ish landmasses near the capital.		Strong and uncapped impact of distance in all eras, making the AI highly reluctant to settle e.g. the New World on Terra.
Rationale	A river (or oasis) without resources is rare, but can be worth settling on (esp. with Flood Plains). About the impact of distance: Need to be careful not to revive the AI habit of planting colonies on a far edge of foreign territory. I think that used to happen in Vanilla Civ 4	

	and in Civ 3.
Fixed a likely bug in the K-Mod evaluation of additional copies of a resource near a city site.	Especially extra copies of strategic resources had been way overrated.
Utility of happiness/health/strategic resources (i.e. not the utility from yield) halved when the resource can't be connected yet. Further reduced if there is already an improved copy available.	Utility computed as if the resource could be connected. Unimproved copies not taken into account. This leads the AI to aggressively settle near any Oil sources when Oil isn't yet workable.
Reduced the penalty for bad tiles near a city site by subtracting a multiple of the number of bad tiles rather than dividing by it. Decreased the base utility that the found value starts at.	Sites with 10 or more bad tiles practically never settled. The bad-tiles check seems to be the main guard against settling land that isn't (ever) worth settling.
Utility reduced when a site has very little food; to the point of making the AI unwilling to settle completely dry areas. The penalty is not applied to resource trade values though, so the AI could e.g. settle in the middle of a desert in order to get access to Oil.	The code that checks the available food only takes into account food on resource tiles, along the lines of "the Grassland Cow will allow this city to work its Plains Hill Gold" but doesn't take into account e.g. regular Grassland Farms, and the low-food penalty is rather moderate. It applies to the trade values of resources though (although trading/ sharing of resources does not require the tile to be worked).
Snow, desert and tundra hills counted as bad tiles. Ocean tiles count as half bad.	Hill and ocean tiles are never counted as bad (nor half bad).
When a resource enables units, the AI power values (from XML) of those units are factored into the utility value of the resource (which, in turn, factors into found value).	Evaluates the unlocked units only based on how useful they are compared with other currently available units. Tends to overrate Iron and underrate Oil and Uranium (once they can be worked).
Increased the utility counted for yields from bonus resources, especially if these yields are provided from a small number of powerful resources.	Utility is assigned based on the total yield from bonus resources (assuming they're all improved and worked); the number of resources doesn't matter in this context.
The per-tile utility values are decreased by a constant, then sorted in descending order and multiplied by decreasing weights. The total is multiplied by a normalization factor to keep it on a similar scale as in BtS. The special penalty for bad tiles is much lower than in BtS.	The per-tile values are mostly based on tile yields. The values are summed up without weights (or with uniform weights if you will). A special penalty is applied for "bad" tiles toward the end of the found value computation.
Rationale	The subtracted constant represents the food consumed by the citizen working the tile and largely replaces the bad-tile penalty. The food consumption is the reason why e.g. a yield of 2 food and 2 production is about twice as good as 2 food and 1 production. The weights give the AI a tendency to prefer a few good tiles over a lot of mediocre (or marginal) ones. A few powerful resources (e.g. Gold, Pig) are preferable to several weak resources (e.g. Silk, Fur) because, in the latter case, it takes longer and costs more food to grow the city enough to work all the resources. Also, while cities with few worthwhile tiles aren't very profitable, they also don't cost much maintenance because the population stays small.
Tbd.	Shouldn't count bad tiles at all. The "special" yields from resources are also dubious; should be possible to cover these through the tile weight distribution. The "taken" tiles count also seems redundant. Try this:

Count for each tile in the city radius (and the city center) a yield vector that is the sum |

of the nature yield and, if applicable, resource improvement yield. Subtract the 2 food consumed per citizen. Compute from the yield vector a tile yield value that should also include a base value to account for a run-of-the-mill improvement and (negative) maintenance and civic upkeep paid for the extra citizen. (Predicting the improvement type gets too far into the weeds I think.) Also compute a food surplus value for each tile – or perhaps better two values: a natural food surplus and an improved surplus that assumes Farm and Lighthouse if those are legal. Perhaps just assume another +1 improved surplus in the Industrial era to account for Biology and food corporations. From the tile yield values, culture modifiers, the sequence of decreasing weights and the per-tile food surplus values, `AIFoundValue::sumUpPlotValues` needs to compute the tile value, essentially simulating the growth of the city population, applying penalties when the cumulative food surplus runs low or when the city relies on food from culturally contested tiles. This should get rid of all the "special" variables and the bad tile and taken tile counting.

When there is no easy means of generating culture, unowned tiles in the outer ring and all foreign owned tiles should be penalized. This could be accomplished by changing the criterion by which tile values are sorted before computing the weighted sum. E.g. the first two positions could be reserved for tiles that can be worked without expanding borders. Sorting should also be biased toward a high food surplus. Will need a (private) CityPlot class for the sort criterion; should also be useful for tidying up the main tile evaluation loop in `AIFoundValue::evaluate`.

Apart from the overall yield value, the weighted sum could also provide (a rough guess at) the maximal city size and the cumulative yields upon reaching that size. Those yields could be used to reward cities that specialize on one type of yield. The maximal city size should also be useful for evaluating health.

Utility threshold for founding a city increases throughout the second half of the game. Threshold lowered based on the number of owned corporate HQs.	Increases only based on number-of-cities maintenance, not game progress. Threshold lowered based on total city maintenance (to prevent overexpansion), which includes corporation maintenance and can lead to very high thresholds once corporations have spread.
Rationale	Toward the end of the game, too little time remains for cities to become profitable and amortize. Corporations can cause high city maintenance before the Modern era, i.e. not that late in the game. At that point, high city maintenance shouldn't worry the AI too much because the corporations bring in high yields that can be turned into gold if necessary.
AdvCiv	*BBAI*
Reduced the utility assigned for access to the sea, especially when the site doesn't offer much production, when already having many coastal cities and when playing on Pangaea. Nothing to encourage the AI to found at least 25% of its cities at a coast.	Code added to encourage coastal cities for more AI shipbuilding but doesn't take into account how much production these cities would actually have. Also adds a clause that gives a substantial extra boost to coastal sites when fewer than 25% of the owned cities (rounded down) are coastal. This can lead to coastal cities in totally awful sites.
Rationale	The current code may still encourage more AI coastal cities than would be rational, and that's OK with me if it helps make naval units viable. But I see no reason to have every AI civ found at least a few coastal cities – being entirely landlocked isn't a particular disadvantage.

Tbd.	Taking the map script name into account could help in other parts of the AI too.	
AdvCiv		*K-Mod*
Bug fixed: Tech requirement for removing Jungle had not been taken into account. Great reluctance to settle sites that will lose food due to bad health during the Ancient and Classical era. No change for other eras (except for the bugfix).		The AI happily settles its second city in the middle of Jungle.
Some extra utility for chopping opportunities, utility from good health capped.		Chopping only accounted for indirectly through good health.
See also	119 allows chopping only on owned tiles.	
When computing the total potential production rate at a city site, count 1 production for non-Hill land tiles only if a worthwhile improvement that grants production (i.e. a Workshop with +2 production) can actually be built there. 2 production per Hill (no change).		Count 1 production for any non-Hill land tile, even a Peak and regardless of available technology.
Some utility is counted for tiles in the radii of other cities, i.e. tiles that could be stolen from rival cities or shared with a friendly city. Such tiles are not counted as bad tiles. Stolen tiles: Tile utility is reduced through essentially the same formula that K-Mod uses for tiles that have a rival owner but aren't in any city radius. I'm making that formula more pessimistic overall and especially for tiles in a rival city radius because the rival city owner is more likely to focus on culture when there is a contested workable tile. A second (new) formula accounts for potential diplomatic (or not so diplomatic) consequences of stealing tiles. This formula is personality-based (Protective trait, power threshold for limited war). Shared tiles: I'm counting only those tiles that are going to be assigned to the new city, and only tiles that the old city is probably not going to need in the medium term (i.e. not until it grows three more times). Even then, the old city is normally going to miss the tile at some point, and I'm reducing the yield value counted for the new city to about 40% to account for this. The tile is counted fully for deciding whether the new city is going to have sufficient food and production. I haven't changed the AI code that assigns shared tiles to cities.		Tiles within the radius of a city (rival or friendly) are counted as "bad tiles" and otherwise disregarded. For tiles outside city radii that are within rival borders, tile utility is decreased (a bit, say, by 25%) based on the amount of foreign tile culture. The AI can place cities close to each other, it just doesn't expect the new city to win any contested tiles. When the radii of two cities owned by the same AI civ overlap, the tiles are assigned based on step distance with orthogonal vs. diagonal as a tiebreaker. K-Mod adds a routine for reassigning tiles when close to a culture victory (`CvCityAI::AI_getCityImportance`).
Rationale	This should mostly allow the AI to settle map regions with abundant resources (or floodplains) more densely than regions with normal or scarce resources, in particular in Earth scenarios (which tend to place a lot of resources in Europe). Multiple cities can start working the good tiles faster than a single city can.	

	K-Mod comment in `CvPlayerA::AI_foundValue_bulk`: "*it kind of sucks that no value is counted for taken tiles. Tile sharing / stealing should be allowed.*" As for counting bad tiles: Tiles in a city radius that the new city is very unlikely to win are really bad tiles, but it's difficult to count them as such because of the structure of the code. The bad-tile counting is a kludge anyway; I'm not sure that it's needed anymore at all (or maybe just as a shortcut to improve performance). karadoc on the bad-tile clause (`CvPlayerA::AI_foundValue_bulk`): "*this final condition is... not something I intend to keep permanently.*"
Tbd.	Ideally, the formula for diplomatic consequences of stolen tiles should check whether the rival is a military threat.
See also	099b (culture decay) makes it a bit easier to steal tiles, especially those outside a city radius.

The Financial trait is factored into the predicted tile yield.	Only done for river tiles, not for coastal tiles. AI civs with a unique naval unit or coastal building have a preference for coastal spots, but the Financial trait has no such effect.
See also	The nerfed Financial trait (908a) still benefits coastal tiles.
AI leaders with Growth flavor are more inclined to settle at a plot distance of exactly 6 tiles from the nearest city than leaders without Growth flavor.	Growth flavor biases the AI toward settling at a plot distance of 6 or more(!) from the nearest city. Plot distance is computed as max{deltaX,deltaY} + floor(0.5 * min{deltaX,deltaY}).
Rationale	Growth flavor should mean that a leader likes big cities, so the city radii shouldn't overlap much. Cities that are 6 tiles apart don't overlap; shouldn't encourage even larger distances that may lead to unworkable tiles (or a another city getting crammed in later on) and high distance maintenance in the early game.
When a tile connects two seas (non-Lake water areas), found value is increased based on the size of the smaller of the two seas.	The possibility of a canal does not affect found value. The AI considers canals when placing Forts but only under narrow conditions and not on workable tiles, meaning that the AI can't establish a canal after settling in the city radius around the tile that allows for a canal.
See also	121 (partly) deals with the Worker AI for Forts as canals
Tbd.	This only works if the two seas are fully separated from each other by land or (through change 030) ice. Will have to call the pathfinder to identify mere shortcuts like in Panama or at Suez. Could do it as follows: At the start of the game let `CvGameAI` compute a canal value for every land tile. It would be better to do this e.g. once per turn and for each team independently based on the revealed portion of the map and OB agreements, but I think this would be too costly to compute. The canal value of a tile T should equal the number of movement points saved by ships passing through a canal in T. Algorithm: Iterate over the tiles adjacent to T in clockwise order (very cheap to do). Make a list of all water tiles encountered directly after a land tile. In most cases, that list is going to be empty or a singleton; this means the canal value is 0 or very small; set it to 0. If the length of the list is greater than 1 (maximum: 4), i.e. if there are adjacent water tiles separated by land, run the pathfinder on each pair of tiles in the list. Set an upper bound for the path length somewhere between 30 and 40 (won't care if the shortcut is even greater) for efficiency, and let the pathfinder only worry about water, land and impassable tiles. Store the maximum of the computed path lengths as the canal value. Use the canal value for the AI placement of cities and Forts.

Deleted a block of BtS code that had increased AI found values by 20% to 50% unless the city was going to be the first colony on a landmass where a rival civ already had a city.	
Rationale	This was apparently supposed to discourage AI colonies on continents dominated by rivals. Such cities are indeed difficult to hold onto. However, the checks were much too coarse (a single city owned by a rival – perhaps even a friendly one – should not discourage colonization) and the positive-reinforcement approach lead to a high bottom for found values on the home continent, making the AI willing to settle anywhere where an abundance of outright "bad" tiles didn't prevent it. Also, the code did little to drop the found values of exposed colonies below the found threshold, meaning that they were merely deprioritized compared with local cities. That's something that the K-Mod evaluation for distances and empire shape will also accomplish.

Deleted a block of K-Mod code that was trying to steer starting locations toward an even distribution of players among landmasses.	
Rationale	This should be the responsibility of `CvPlayer::findStartingAreas`.

AdvCiv	*BtS*
Count only major landmasses when calculating the target distance between starting sites.	All landmasses are counted, up to a maximum of the player count. This tends to lead to unachievable target distances on Huge maps.

Rationale	Looks like an oversight.
See also	[137](137) adjusts the target distance to the greater default player counts in AdvCiv.

AdvCiv	*BtS/ K-Mod*
Deleted the "greed" personality modifier.	A BtS comment said that "*Greedy founding means getting the best possible sites - fitting maximum resources into the fat cross.*" K-Mod introduced modifiers "easy culture" and "ambitious" that, taken together, are based on the same leader personality values and affect the found value computation in similar ways.

Tbd.	"Greed" in the sense of cannibalizing other potential city sites could make sense. For situations when there is abundant space for expansion; possibly also based on leader personality.
See also	For the latest version of the deleted code (greed and the deletions mentioned above), see this Git commit.

Lower threshold for founding cities when nearing a Domination victory.	
Rationale	Cities on marginal land can be a significant contribution toward meeting the target land percentage, which is usually the tougher of the two Domination conditions.
See also	[115b](115b) makes the AI more willing to grow city populations when approaching a Diplo victory.

I've done away with all or nearly all fog-of-war cheats in the evaluation of city sites.	Resources had been handled correctly in K-Mod I think (probably not in BtS), but the fog of war had been ignored for tile ownership. This also meant that city sites recommended to human players gave away AI cities in the fog of war. The BUFFY mod (not included in K-Mod) had shown recommendations only on actively visible tiles, (presumably) to fix this info leak.

AdvCiv	K-Mod
When normalizing starting sites (i.e. when determining whether a site needs additional resources), some value is counted for Tribal Villages in the starting city radius.	Tribal Villages aren't accounted for at all.
See also	027 will deal with Tribal Villages beyond the starting city radius. Also allows removing Tribal Villages in order to free up a starting site.

031b	AI trains Settlers only probabilistically when city sites are poor	
AdvCiv		*BtS/BBAI*(?)
AI governors compute a priority value for training Settlers based on the found value of the best city site, the current found value threshold (which is, among other factors, based on expenses) and the game options Always War and Always Peace. Protective leaders and voluntary vassals prioritize Settlers more highly. When all necessary requirements (no change) for training a Settler are met, the probability of training a Settler is based on the priority value. During the first 100 turns (Normal speed), the probability is typically 100%; so the change tends to matter most between turn 100 and 150.		When the conditions for training a Settler are met (nothing very urgent to do; at least one city site whose found value is above the threshold; OK for city to halt growth), AI governors train Settlers with 100% probability regardless of the quality of the city sites. Exception: In defensive wars, the probability is only 50%.
See also	CFC user keldath has pointed out to me here (and in a private conversation) that the AI tends to expand too recklessly.	
Rationale	In part as a consequence of my changes to AI found value (id 031), the AI will claim any land where a city can develop if it has relatively few cities (and no better land is available); I think that's working as intended. However, the AI shouldn't *be quick* to claim land that is barely worth settling. Delaying a Settler for some 5-10 turns can open up new avenues, e.g. through revealing additional tiles or resources. Or perhaps some other civ will claim the site, one whose capital is closer to it; that's also fair enough.	
AI may decide to cancel the production of a Settler when there is no city site.		Unless in a defensive war or financial trouble, Settlers are always completed once started. The Settler is then disbanded after 20 turns unless a city site becomes available.

031c	Log file for AI found value computation	
AdvCiv		*BBAI/K-Mod*

If the DLL is compiled with the `LOG_FOUND_VALUE` flag, then a breakdown of the found value computation is written to the BBAI log whenever an AI civ founds a city. In addition, a breakdown for the next best adjacent city site and for the next best city site is written to the log file. A breakdown is also written when placing a starting location and when "normalizing" (cf. [108]) a starting location.	BBAI has added a global function logBBAI for AI logging. It doesn't cover the AI found value computation though. K-Mod added a little bit of information that enters into the found value computation (e.g. about deadlocked resource tiles) to the tile hover text in Debug mode.

Rationale	It's a complex computation and, so far, my only means of testing it has been the debugger, and that only worked with a savegame from right before the founding of the city.
See also	[003]: Moved the found value computation into a separate class and refactored it. The logging code was crucial for testing those changes.
Tbd.	I've disabled the display of deadlocked resources display in hover text because I wanted to move that computation into `CitySiteEvaluator.cpp` (it isn't used elsewhere), and that has made it awkward to expose the function to CvGameTextMgr.

031d	AI exploration near city sites	
AdvCiv		*BtS*
When moving an exploration unit, the AI prioritizes unrevealed tiles near its (preliminary) city sites over other unrevealed tiles.		City sites play no role in exploration.
Rationale	The AI found value logs showed that the AI is sometimes placed its second city one or two tiles away from a superior location due to unrevealed tiles – even when starting with a free Scout. Also, prioritizing exploration near the capital tends to delay meetings with rival players, especially with human players, which is in my opinion better for overall gameplay.	

031e	`bNormalize` flag for city site evaluation function

Mostly a refactoring change. When placing starting sites and when deciding how to normalize starting sites, found-city values need to be computed with some special provisions; not quite the same in both cases. BtS had a flag `bStartingLoc` and normalization was determined based on that and by checking if starting locations had already been assigned. Meaning that, by default, the `bStartingLoc` behavior had applied also when normalizing.

I also made some minor changes to the normalizing vs. start-placing behavior.

Rationale	Better to be explicit about this.
See also	This change also allows found values to be computed "as if normalizing" before starting sites have been assigned. That might be helpful for [027] (though I don't think I'll make use of it after all).

031f	Adjustments to city site evaluation in scenarios

When evaluating city sites for a scenario with randomized starting locations (e.g. Africa), some adjustments are applied in order to make up for the lack of a normalization step.

Rationale	Without normalization as a fallback, food resources and freshwater need to be given higher priority.

Tbd.	Perhaps needs to be more pronuonced. I'm still getting starts with just one weak resource or none sometimes in e.g. the Europe scenario. That said, all the scenarios based on the Earth's continents have regions with very sparse resources.

032	Signing a dual deal when there already is one causes turns-to-cancel to be reset	
AdvCiv		*BtS*
When a peace treaty, open borders or defensive pact is signed between two civs that already have such a deal, then the turns-to-cancel are reset to 10 turns. This can happen through a diplo vote, or (peace treaty only) sponsored war (due to change 146) or city trade (change advc.ctr). I don't think gifts and demands are possible when there is already a peace treaty. OB and DP votes are nevertheless only allowed (as in BtS) when some pair of full members doesn't already have an OB or DP deal. I.e. the vote can't be put forth just to reset the turns-to-cancel of existing deals. (Nor can it be used just to force deals with non-full voting members – although this BtS restriction is a bit curious.)		Signing a peace treaty has no effect if there is already a peace treaty. Same for OB, DP.
Rationale	More intuitive this way. Otherwise, e.g. the target of a UN peace proposal could get attacked just one turn after the vote. Or Open Borders could be canceled shortly after they've been enforced by the Apostolic Palace.	

033	Changes to naval blockades, Privateer	
Tbd.	Try excluding Privateers from "show enemy moves". When are messages about pillaging and Privateer combat shown to the player then? Combat log? Is it still possible to deduce the owner of a Privateer from the turn order? Try delaying these messages too. The movement bonus from circumnavigation can also reveal a Privateer's identity; I intend to replace that ability with some trade route bonus. Or give up the secrecy; seems historically dubious anyway. (Wikipedia: "The convention was a vessel must hoist her true colors before firing the first shot.")	
See also	007 reveals the owners of Privateers when in Debug mode 905b gives Privateer (and Frigate) +1 speed	
AdvCiv		*BBAI*
When collecting gold from Privateers, compute the plunder range based on path distance (e.g. not across an isthmus).		BtS does it all based on air distance. BBAI uses path distance for blockaded tiles, but jdog seems to have forgotten to change the plunder code as well.
Rationale	Or they had been worried about performance, but I'm only checking cities adjacent to blockaded plots, which should be quick.	
See also	030 (ice-locked water areas) depends on this change. 124: Can use blockade to let trade pass through hostile tiles.	

AdvCiv	BtS (BBAI and K-Mod made some changes to the Privateer AI, but the basics are unchanged)
When selecting a city to plunder, the AI takes its leader's attitude toward the city owner and the tech known to the city owner into account. The AI doesn't use Privateers against ships and improvements of partners. What a "partner" is in this context depends on the AI personality. If the attitude is equal to `DeclareWarThemRefuse AttitudeThreshold` or worse, AI Privateers will attack. This threshold is Pleased, Cautious or Annoyed for all leaders except Catherine (Friendly).	AI Privateers behave largely like Barbarians and treat all civs except the Privateer's team alike.

Rationale	It's not smart to use Privateers against one's partners, especially not for attacks and pillaging. An attack puts the Privateer at risk for no gain and the gold from pillaging water improvements is negligible. Plundering a partner should be OK when there is no one else to plunder.
Tbd.	Privateer attacks on ships should have some upside for the Privateer owner. Stalking ships is fun but just not rewarding. A ransom mechanism would be nice because it would also curb the losses of the owner of the attacked ship but too much work to implement at this point, and no other mod seems to have implemented this either. A chance to steal the ship (in a damaged state, and any cargo transformed into a single Worker)? Captured Caravels wouldn't be very useful though ...
See also	130v makes vassals Friendly toward their master.

Civs in a vassal-master relationship can't pillage each other's improvements with Privateers, can't block tiles from being worked (`CvUnit:: canSiege`) and can't blockade each other's cities. They can still attack each other's ships through Privateers, and no restrictions apply to vassals of the same master.	Privateers can attack, pillage and plunder everything not owned by the Privateer's team.
The AI never attacks ships of its vassals or master. (That's the idea anyway. Seems difficult to guarantee this ...)	AI Privateers attack anything that comes near them.

Rationale	About capitulated vassals, see below. Voluntary vassals plundering their master might be OK, but a master plundering its vassals (voluntary or not) is jarring; the master is supposed to protect the vassal. An AI change wouldn't stop a human master from harassing his/her vassals, so this has to be prevented on the rules level.
	I've considered changing `CvUnit::isEnemy` and `getCombatOwner` so that Privateers of vassal/master behave as if they're on the same team, and revealing the Privateer owners' identities between vassal and master. This would entail the following: • can't attack each other and can coexist in the same tile • can heal in each other's territory and enter each other's cities • can't blockade each other • moves shown as friendly • Privateers not seen as a threat by the AI
	This would be a pretty clean approach, but goes a bit too far I think. Implausible that everyone on the vassal's and master's side knows a Privateer's identity, but other civs have no clue. Submarines aren't revealed between vassal and master either.
	If the owner if a vassal/master Privateer is to remain secret, then it has to remain possible (for humans) to attack Privateers; otherwise one could tell who the owner is. And if Privateers can be attacked, they should also be allowed to fight back, so, no restrictions on Privateer combat. It's a little strange that combat is possible, but not pillaging; I guess there's no perfect solution for this.
Credits	Cruiser76 raised the issue here on CFC.
See also	123e makes it impossible to plunder Barbarian cities
Tbd.	When using a Privateer to attack a unit that the Privateer owner is not at war with, and hovering for combat odds, the mouse cursor shows a white circle. I think it should be a red circle like for all other attacks. This isn't a result of my changes; it's a white circle in BtS too.

Capitulated vassals only train Privateers if they somehow have no other general purpose naval unit. All civs stop training Privateers once the game enters the Industrial era.	Capitulation doesn't affect the training and behavior of AI Privateers. There is a unit "viability" check (`CvPlayerAI::AI_calculateUnitAIViability`) to prevent the production of outdated units, but that's based only on units that the same civ can train, not the global tech level.

See also	130v makes capitulated vassals subservient to their master in most regards.
Rationale	Capitulated vassals shouldn't even indirectly interfere with their master's goals. Don't want them to use Privateers against civs that the master might like (with a human master, one can never tell) and don't want a blockading Privateer to affect the master indirectly e.g. by cutting off some important sealane.

The Privateer's plunder ability is listed in Civilopedia and help text.	Only mentioned on the BtS concept page about trade blockades. For the Privateer, only the hidden nationality ability is listed. A feat message ("Congratulations, you have trained your first Privateer ...") also explains plundering, but that's a one-time notice.

Rationale	Technically, the Privateer doesn't have a special plunder ability. Any ship can collect plunder if it can manage to blockade a city without declaring war. This rule is very obscure though; need to list plundering as a separate ability. Also, when a player wonders why a Privateer can't plunder a vassal, that player may look up the Privateer in Civilopedia, and there it'll now say that Privateers have the ability to plunder *rival* cities.
AI civs don't blockade Barbarian cities.	Not sure if the BtS code actually targets Barbarian cities for blockades or only ends up blockading them opportunistically while bombarding.
Rationale	No gain in starving Barbarian cities.
Tbd.	Naval bombardments of Barbarian cities might still happen, and then the AI would also blockade. There's no point in this because Barbarian cities normally only have building defense, and by the time Frigates become available, most AI land units ignore building defense.
Only Privateers can blockade at peacetime.	All warships can blockade inside the Open Borders of other civs. This only affects the trade of the ship owner's war enemies (if any).
Rationale	Confusing; makes it look like the tile owner is being blockaded. Also not realistic that a civ that has Open Borders with both sides of a war would allow a blockade in its waters.
Non-Privateer units can't blockade at all if not at war with any civ.	
Rationale	Blocking Barbarian trade doesn't make sense. The advantage is that blockading units will be woken up (with the change below) once a war ends (well, once all wars end). It's easy to forget about blockading units after making peace.
When the owner of a tile changes, blockading units in the tile are only woken up if the blockade is no longer legal.	Can e.g. prevent a Privateer from collecting gold.
Blockading units are awoken at the end of a turn if the blockade is no longer legal; e.g. because the owner of the current tile has capitulated, or (see above) no longer fighting any war.	Will blockade (to no effect?) indefinitely unless manually woken up.
Added a function `CvUnit::blockadeRange` for code shared by `updatePlunder`, `collectBlockadeGold` and `CvGame::updateColoredPlots`. This should also address some minor inconsistencies between them. Also replaced some duplicate/ inconsistent code with calls to `CvUnit::canPlunder`. Now using a pathfinder function written for 104b for computing the tiles affected by a blockade because that function has a range limit (whereas the BtS function `CvMap::calculatePathDistance` is quite slow when tiles are unreachable) and can handle movement restrictions of non-oceangoing ships. Removed some unnecessary updates of plot groups and trade routes. This change mostly eliminates the delay after starting/ending a blockade with a non-Privateer unit; doesn't really help with Privateers though.	
See also	Without 003m (which caches `atWarCount`), this could be slightly slow.

034	Temporary Open Borders after war (optional through XML, **disabled** by default)

AdvCiv	BtS
When a war ends, an Open Borders (OB) agreement is automatically signed between the former belligerents. Units aren't bumped until that agreement ends after a configurable number of turns (recommended: 3 to 5). It's a special OB agreement, internally called a "disengagement" agreement, that does not affect trade and diplomacy; only a right of passage for units. Details about the temporary OB agreement: • No movement bonus from roads and railroads. • No risk of getting caught for idle and moving Spies. • Canceled when a proper OB agreement is signed, or when a Vassal Agreement or Permanent Alliance is signed. • Shown as "Open Borders (x turns)" on the scoreboard and in other help text. • The turn on which peace is made counts as the first turn. When it says "1 turn" remains, then the agreement is canceled at the end of the round. • Unaffected by embargoes.	When a war ends, units are immediately bumped out of rival territory, teleporting to the nearest tile that they can legally occupy. The AI is not programmed to anticipate this.

Rationale	To allow the AI to reposition its units after a war, in particular, to prevent large AI stacks from getting stranded. This happens quite regularly when the AI conquers a remote city, and it can be very harmful for the AI because it still counts on those units when when deciding on war/peace against a third party and when calculating needed floating defenders. The change also reduces unit bumping, which is a pretty strange mechanism. The ability to explore the territory of a former war enemy just after the war ends is highly immersion-breaking though. It's worse than I had anticipated before implementing this change.
Config	Increase `DISENGAGE_LENGTH` in `GlobalDefines_advc.xml` to enable this change.
Tbd.	Not properly tested. The "You have made peace with" message (`CvTeam::makePeace`) should also mention the Open Borders.
See also	035 (also disabled) would also address the problem of stranded units. 099b (tile culture decay) addresses it a little bit. 046: AI improvements for units stranded on a different landmass and changes to the bump algorithm.

035	While at peace, civs own all tiles that no other civ could work

Config	Currently **disabled** (including the AI changes) through the `OWN_EXCLUSIVE_RADIUS` switch in `GlobalDefines_advc.xml`. Has been disabled for some time, but, as of v0.96, it still seems to work correctly as reported [here]().
See also	[099b]() replaces 035. If 035 is enabled, then the `CITY_RADIUS_DECAY` part of 099b should arguably be disabled. Revolt chance ([101]()) is adjusted to 035 if it's enabled.
Tbd.	Try letting civs only own those tiles in the exclusive radius where they're somewhat close to having the majority culture; e.g. set a multiplier $m := 75\%$ and let civ *A* own tile *T* in its city radius if culture(*A*,*T*) > m*culture(*B*,*T*) for every civ *B* that does not have T in a city radius, and culture(*A*,*T*)>culture(*C*,*T*) for every civ *C*≠*A* that does have *T* in a city radius. I guess inner and outer radius should be distinguished too. Perhaps too complicated, but, other than that, it might play better than the decay in 099b.

AdvCiv	BtS
A civ can only own a tile when that tile is either within the (workable) radius of one of the civ's cities, or when it's not within the radius of any cities that the civ is at peace with. I.e., at peace, cities own all tiles that they could work and that no rival city could work, and the highest culture value only matters for tiles in overlapping city radii and for tiles that are in no city radius. At war, tile culture matters on all tiles contested between the war parties, and thus some tiles may flip when war is declared and when peace is made. A unit that was about to cross the border can get bumped back upon declaring war. Minor AI changes (more below) Forts: Don't build them on tiles that will flip when at war. Pathfinder: When moving to hostile territory and war is not yet declared, try to avoid tiles that are going to flip to the enemy upon DoW, and prefer enemy tiles that are going to become friendly.	The civ with the highest tile culture owns a tile so long as that tile is within the culture range of one of the civ's cities (up to 6 tiles depending the city's culture level). Thus, tiles within a city's radius can belong to a civ that is too far away to work the tile, meaning that no civ can work it. War/ peace does not affect tile ownership.

Rationale	It's implausible that tiles near a city can't be worked by anyone. At least, this shouldn't happen as commonly as it does in BtS. It's a mechanism that encourages culture, but it does so mostly through punishment (loss of workable tiles). A civ with high culture gains tiles, but it's not much of a reward because that civ can't work them; it only gets bonus resources and a tactical advantage, which is somewhat neutralized by the diplo penalty for stealing tiles. My change preserves the tactical advantage.

Border tensions: When not at war, stolen tiles in overlapping city radii count 50% more than in BtS. Tiles that would flip when at war add to border tensions; the civ that would own them when at war is mad about them.	

Rationale	The rules change result in fewer stolen tiles, but I don't want less border tension. The diplo penalty for tiles that flip when at war works against civs with low culture (good), whereas the diplo penalty for stolen tiles works against civs with high culture (not good, as in BtS).
See also	[147]() also changes the border tensions formula.

AI found value: Don't worry much about foreign culture on tiles near a city site, so long as those tiles are not within the radius of a foreign city. Take into account which tiles will flip when settling near a rival city.	
Rationale	The AI should still avoid foreign culture a bit because tiles around the city will flip when the owner of the foreign culture founds a city nearby or when war is declared (making the city difficult to defend). Foreign culture will also add to border tensions (see above), and can cause the city to revolt.
See also	031 makes other changes to found value.
UWAI: Reduced utility penalty from culture-pressure penalty (`GreedForAssets` aspect).

Tiles lost by war enemies are counted as "lost assets", making the AI more reluctant to make peace when this will flip tiles to the enemy. Flipped tiles could also motivate the AI to start a war that otherwise looks like a stalemate. | |
| *Rationale* | A peace treaty practically cedes the tiles around any lost cities to the conqueror. The AI needs to be reluctant to do that and factor it into reparations. |

036	AI changes concerning resource trade
See also	133: Cancelation of resource deals
074 "You must be joking" resources excluded from the trade screen
210d: Alert about AI-AI resource trades
912c changes the ability of the Hereditary Rule civic because 036 would otherwise make too many happiness resources available for trade.
073 makes some changes to the "Resources" tab of the Foreign Advisor screen, in part, to accommodate trade with non-surplus resources better.

CFC post praising/ defending the AdvCiv resource trade AI changes. |

AdvCiv	BtS
The AI takes into account the current happiness and health (and anger and bad health) of its cities and buildings like Grocer and Forge when evaluating resources for trade or the placement of cities and improvements (found and build value).	The AI does a detailed evaluation of strategic resources but assigns the same utility value to every health and happiness resource except that surplus resources are valued less.
The evaluation for trade differs a bit from the evaluation for found and build value. Since a trade can be canceled after 10 turns, the resource needs to be useful in the short term.	A single "`baseBonusVal`". The AI trades for strategic resources long before it needs them.
The AI refuses to accept resources with low utility in trade: "We would have nothing to gain".	Not a matter of utility. The AI accepts any resource that it doesn't already have, and any resources consumed by its corporations.
The AI is willing to trade away even non-surplus resources when it doesn't have a use for them. (But the AI still doesn't wheel and deal, i.e. never both exports and imports the same resource.)	The AI only trades away resources that it owns (directly; not through import) multiple copies of.
E.g. in the early game, the AI typically refuses to accept health resources in trade and is willing to trade away health resources that it has only one copy of.	
The AI knows how to buy resources from other AI civs for gold per turn and can even out resource-	Only trades one resource for one resource.

for-resource trades with gold per turn.	
The price in gold per turn that the AI charges for a resource is computed based on the utility for the recipient and a "market value". The utility of the recipient is computed from the number of its cities, and the happiness and health in those cities if a sufficient portion of the recipient's territory is revealed to the AI. The market value is based on the number of other potential buyers and the number of their cities. The market value tends to be quite a bit lower than the recipient's utility.	Based only on the city count of the seller and the recipient. Resources sold between large civs tend to be overpriced.
Relaxed AI attitude threshold for trading away strategic resources when the recipient already has access to the respective resource and only needs additional copies for a corporation.	The leader-specific threshold applies regardless of whether the recipient already has the resource.

Rationale	The more flexible AI behavior creates a larger market for resources, allowing civs that lack resources to grow their cities through reasonably priced imports. Should make it more feasible to grow tall without growing wide.
	The utility of the seller does not factor into the price for resources because the AI still only trades away resources that it doesn't have much of a use for.
	I've considered allowing resources in peace deals, but that's not workable. Too difficult to predict whether there is going to be a trade connection, and the civ providing the resources could just pillage its own roads to get out of the deal. Might also cause problems outside the SDK.

AdvCiv	BUG (Exotic Foreign Advisor)
In the list of resources that the AI is willing to import from the player (Resources tab of the Foreign Advisor), surplus bonuses are listed before bonuses that the player has only one of.	BUG's Foreign Advisor screen (enabled by default; not sure if it can be disabled) lists resources that other civs are willing to import or export on a tab "Resources". The lists are ordered by the (meaningless) internal ids of the resources.

Rationale	It's preferable to trade away surplus resources, so these should be distinguished from other resources that the AI is willing to import. This is not directly related to the AdvCiv changes to AI trade behavior (in BtS, humans can export non-surplus resources as well).
Tbd.	I've tried to highlight the surplus resources in the import list somehow, but that's difficult to do because the lists are shown through BUG's `IconGrid` class, which uses `addMultiListControlGFC`, and that function treats every listed icon the same way. Perhaps a graphic could be overlaid like it's done under the BUG comment "*add the circles behind the amounts*" in `CvExoticForeignAdvisor.py`.

AdvCiv	BtS
The AI makes up to about one third of its total commerce available for trade as gold per turn. The current income also no longer factors into the amount of cash that the AI is willing to trade. Instead assume an income equal to 20% of the total commerce; this usually leads to more cash	The amount of gold traded per turn is capped at the current per-turn income (commerce times gold slider position minus expenses). This cap is usually a one-digit number or a small two-digit number. Often, it's negative, meaning that the AI is unwilling to trade away any gold per turn. The amount of cash that the AI is willing to trade

for trade than in BtS.	is partially based on the current income.

Rationale	Now that the AI knows how to evaluate resources, there's no harm in making gold available for resource trades. This allows small AI civs to take advantage of the resource market. It also allows large civs to sell most of their surplus resources, but the prices, especially those paid by small civs, are low, so this change helps small civs more than large ones. A low income shouldn't prevent the AI from trading; doesn't imply financial trouble.
See also	104w lets UWAI decide how much gold the AI should be willing to pay for peace. 104m allows the AI to demand gold per turn as tribute. 026 makes the AI willing to trade even more gold in an AI-initiated one-time offer. 550f determines how much gold the AI saves up (gold target). 210e extends the resource trade alert so that it triggers when the AI becomes willing to import a resource in exchange for gold. 131 deals with avoiding strikes during anarchy; the increased trade budget makes the AI more vulnerable to this.

The resource help text on the "Resources" tab of the Foreign Advisor screen shows how much gold per turn the AI will offer or demand for the resource. When hovering over a resource icon in the "Domestic Resources" box, all AI gold offers are shown; when hovering over a resource icon in the "Will Import" column, only the offer of one civ is shown.

See also	073 changes the headings on the "Resources" tab.
Rationale	Makes it easier to find the best buyer or supplier.
Tbd.	Optimization of trades is still tedious. Might want to put some mechanism in place to discourage changes, perhaps simply a counter akin to "We fear you're becoming too advanced" that increases whenever a resource trade is signed.

The AI assigns a fixed trade value to gold (per turn, cash).	The trade value depends on whether the AI is in financial trouble.

Rationale	The BtS approach means that the AI receives less gold in trade when in financial trouble. This makes some sense (you're in a bad position to negotiate when you need money badly) but isn't really intuitive. Finances factor into the amount of gold that the AI makes available for trade; that's good enough to prevent broke civs from overspending.
Tbd.	Would be nice to apply a slight adjustment based on the recipient's attitude.

The gold per turn and cash that the AI is willing to trade to another civ is adjusted based on the attitude toward that other civ except when the two civs are at war or when the AI civ is a capitulated vassal.	The amount of gold that the AI is willing to trade is not affected by attitude.

Rationale	When e.g. an AI civ promises that it'll "never trade" with its worst enemy, that civ shouldn't be perfectly willing to pay gold to that enemy (but shouldn't refuse to pay any gold either). This change should also make attitude matter more when tech trading is disabled. Exception for capitulated vassals: Don't want the master or the master's friends to plunder the vassal.

AI memory about cash traded to other civs decays.	The AI keeps track of how much gold it has traded to every other civ. That amount is subtracted from the cash that the AI is willing to trade to another civ. This subtrahend tends to become negligible over time because the AI

		generally becomes willing to trade larger and larger sums of gold as the game progresses.
Rationale	The amount of gold that a civ is willing to trade (prior to subtracting gold-traded memory) can stagnate, and then the BtS AI can stay unwilling to trade any gold indefinitely. Also, pretty much all other AI memory decays in AdvCiv (cf. [130r]).	
When asked to "make this deal work," the AI may add resources to the trade table that the human civ has only one copy of if the human civ doesn't have enough gold per turn available for trade.	Never adds non-surplus resources to the table.	
If the human player is offering resources to the AI, then the AI offers to buy them with gold per turn if the current human income is negative, and otherwise tries offering surplus resources in return and uses gold per turn to balance the deal. May offer non-surplus resources as a final option.	Tries to give surplus resources, then gold per turn.	
Rationale	Had to adjust the counter-propose algorithm to the more flexible AI trade denial behavior (which involved refactoring the entire function).	
	The human income condition leaves it up to players whether they want the AI to pay in gold or in resources: Just change the gold slider for a moment. Resources are often preferable because the AI doesn't pay much gold, but gold can also be preferable, especially when the player doesn't need the resources of the AI. Players shouldn't have to optimize the deal through trial and error then.	
The AI refuses to give away more than 2 non-surplus resources of the same kind (happiness or health) at once, and refuses to accept more than 2 resources of a kind at once.	Capitulated vassals give any number of non-surplus resources to their masters. Other civs never give away non-surplus resources.	
The precomputed resource values are updated after starting or canceling a trade.	No limit on resources accepted at once.	
	Resource values are precomputed (since the BtS expansion) for efficiency and only updated once per turn.	
Rationale	Now that the AI evaluates happiness and health from resources situationally, it must be careful not to buy too many resources at once because the evaluation is done for each resource independently. I.e. the AI knows what e.g. one more happiness is worth, and simply triples that value when three happiness resources are offered (but perhaps needs only one). Similarly, the AI can tell whether it can spare e.g. one non-surplus happiness resource, but it can't tell if it can spare more than that.	
Tbd.	Should call `AI_updateBonusValue` each time that the AI connects a resource to its city network (easier said than done). Currently, an AI civ updates its resource values at the start of its turn, so resources connected during the AI civ's turn aren't taken into account until the next turn of that AI civ.	
Once per round, all AI-AI deals are checked for cases in which two civs A and B sell resources to each other. If one deal is found in which A sells exactly one resource to B, and another in which B sells exactly one resource to A, and both deals can be canceled, then they're canceled and immediately replaced by a resource-for-resource deal in which one side may additionally have to pay some gold.		
Rational	Now that AI civs are able to import resources from other AI civs for gold per turn, resource-for-resource deals have become rather rare because often only one side has a suitable resource available. Sales are harder to read on the Foreign Advisor screen than barters and take up more space. Therefore try to collapse sales into barters.	

When trading one resource for another with a human, the AI never asks for 1 gold per turn in addition.	Not an issue because all health and happiness resources have the same trade value.
Rationale	Adding the 1 gold to the trade table is tedious, and it makes the AI look petty.

AdvCiv	*BtS/K-Mod*
Changed AI tech evaluation to increase the value of bonus improvements with high tile yields (such as Cow Pasture or Corn Farm) and to decrease the value of military units whose bonuses haven't yet been revealed (such as Axeman and Swordsman; multiplier now 60%).	The utility counted for bonus yields is proportional to the yield. (BtS code) Military units requiring unrevealed bonuses are valued at 80% (compared with units whose bonus requirements are fulfilled). (K-Mod code)
Rationale	Since health in the early game is now (correctly) counted as unimportant, the AI tech values of Agriculture and Animal Husbandry need to be adjusted. Prior to this adjustment, the AI often went for Iron Working(!) before Animal Husbandry, even if it had a Cow next to its capital. (Actually, I suspect that this happens in K-Mod too, despite the overestimation of health, but I haven't checked. In unmodded BtS, it's fine.) K-Mod comment on the Axe and Sword evaluation (`AI_techUnitValue`): "*We're quite optimistic... mostly because otherwise we'd risk undervaluing axemen in the early game! (Kludge, sorry.)*" Undervaluing Axeman isn't a problem currently.
Tbd.	Animal Husbandry is still be a bit underappreciated. The strategic value counted for Horse might be too low. The fact that Wheel enables Chariot complicates matters. That said, the evaluation of the resource yields also needs more work. Needs to check whether a new terrain improvement will immediately increase city yields in the very early game.
See also	k146 increases the tech value counted for chopping Forests and Jungles; I've also reduced that effect a bit.

AdvCiv	*BtS*
The AI values surplus resources based on the potential for trading them away, and (as in BtS) based on corporation yields.	Unless consumed by a corporation, a surplus resource is valued at 20% of the first copy. This value is sometimes so low that the AI doesn't connect surplus resources. (Perhaps also due to K-Mod changes to the Worker AI and the evaluation of the first copy; not sure if this is a problem in unmodded BtS.)
Rationale	The value of the first copy shouldn't affect the value of surplus copies; one resource for trade is as good as another. Tradeable strategic resources might be a bit more valuable than luxury and food resources, but not much; difficult to find buyers for strategics. And the AI needs to connect all its (surplus) resources; it's the correct play most of the time, and can't really hurt.
See also	Prior to v0.92, this change was tagged with `advc.121`. Change 121 deals with AI Worker builds.
Set all XML modifiers for the AI trade value of individual resource types to 0.	Some strategic resources have an "AITradeModifier" of 10 to 30% set in XML that increases the trade value assigned by the AI to those resources.
Rationale	The BtS/K-Mod code for evaluating strategic resources should have this covered. (If

	not, it should be amended.)
Config	`Civ4BonusInfosl.xml`

AdvCiv	*Warlords*
The demand-vassal-tribute shortcut demands all resources that the vassal is willing to trade to the master, including those that the master only needs for a corporation.	Players can demand resources from their vassals by holding down the Alt key while clicking on the vassal's name on the scoreboard. This triggers a confirmation popup listing all resources that the vassal has an excess amount of and that the master doesn't have yet. (In Warlords, this was equivalent to all resources that the vassal was willing to trade to the master because there were jno corporations.) If confirmed, the resources are demanded as if through "it's time for your tribute."
Tbd.	Perhaps replace this with a "join us in preparing war" popup. I've actually only discovered this shortcut because I was wondering if karadoc had implemented a shortcut for vassal war preparations.

036b	Changes to resource evaluation not aimed specifically at resource trades
See also	031: Changes to AI city placement

AdvCiv	*K-Mod*
Reduce the the value assigned to strategic resources that enable numerous units or buildings.	Utility values are counted for each enabled unit and building individually. Values are deuced when not all requirements are met and also if a unit compares unfavorably with the best unit already available for the same role.
Rationale	The K-Mod evaluation can't take into account that some of the units enabled by a resource may have the same or similar roles, e.g. Axeman and Swordsman (both enabled by Iron if Copper is unavailable). Similarly, a civ won't always find time to construct any number of wonders.

Put differently, the K-Mod values for Iron, Marble, Stone and some other strategic resources seemed a bit too high. |

037	Prevent masters from stifling their voluntary vassals
Rationale	Voluntary vassals can be difficult to keep, and that's intended. There shouldn't be much that a player can do about this. The game certainly shouldn't reward dirty tricks.
Tbd.	Voluntary vassals should treat "we'd like you to research" and "join us in preparing war" as recommendations and answer "we'll see what we can do" (as opposed to "we'll do our best"). Should also give a noncommittal answer to "why don't you attack."
See also	033 prevents masters from using Privateers against their vassals

AdvCiv	*BtS*
Voluntary vassals lower their trade value to 67% when trading resources with their master. Capitulated vassals charge the full trade value (no change).	50%
Voluntary vassals refuse to trade resources that	Are willing to trade all resources.

they badly need themselves.	
Rationale	Don't want the master to starve the vassal's cities by buying the vassal's crucial resources. Capitulated vassals aren't a concern because players have no reason to sabotage those (they're not going to break away without outside help). Probably no trade discount in BtS because the master can get resources for free through "time for your tribute"; fair enough.
Tbd.	Perhaps put some limit on "time for your tribute".

038	Top Civs popups	
AdvCiv		*BtS*
The "top civs" pop up ("Pliny the Elder has completed his great work: ...") appears for the first time on turn 80, then every 40 turns, but stops appearing once the active player has met more than half of the other civs. The appearance frequency is adjusted to the game speed.		Appears every 50 turns throughout the entire game. No game speed adjustment.
Rationale	The popup interferes with shown foreign moves, and it's somewhat obsoleted by the Statistics screen (demographics). That said, demographics only become available once a civ is met. When starting in isolation or on a continent with few other civs, it's nice to get some sign of intelligent life from the rest of the world.	
Config	`TOP_CIVS_INTERVAL` in `GlobalDefines_advc`. That setting can also be used to disable the popup entirely, as had been the case prior to AdvCiv v0.92. The conditions for showing the popup are easy enough to modify through the `turnChecker` function in `CvTopCivs.py`.	
Tbd.	The map-centering effect (Calendar, Stonehenge) also gets in the way of shown foreign moves; should be moved to the beginning of turns. This may require a variable at `CvPlayer` that is set by `processTech`. Won't have to be stored in savegames.	
The "wealthiest" civs are ranked based on their GNP values (see [004s](#)). The "most advanced" civ is the one with the highest total cost of known techs.		Based on gold in the treasury. "Most advanced" based on the number of known techs.
Rationale	Cash fluctuates a lot; not a meaningful statistic. GNP is more about research than gold, but "wealth" just doesn't really play a role in Civ 4. Research is the most useful information, so it doesn't hurt to have two rankings about that (the rankings all have equal probability of being shown). GNP is hopefully different enough from the tech cost total. The former indicates which civ is coming up in research, the latter says which civ is currently leading. The number of techs can be misleading in the early game. Humans beeline more than the AI does, so the tech count tends to underestimate the human performance. Not good to tell a player who may already be leading in tech that he/she is falling behind.	
The ranking is chosen based on the synchronized PRNG (`SorenRandNum`).		Based on a non-synchronized PRNG, meaning that the outcome can change after reloading a savegame.
Rationale	The chosen ranking has some strategic importance; should be consistent when reloading. OOS errors aren't an issue because top civs popups don't appear in	

	networked multiplayer.
The name of the historian is chosen at random from the two names that I've assigned to each ranking. E.g. Machiavelli and Tacitus for the "most powerful" civs.	The historian is chosen at random from among 11 names regardless of the type of ranking.
Rationale	Just for historical immersion.
Config	I've hardcoded the assignment of historians to rankings in `CvTopCivs.py`.

039	Reparations (payments for peace) are announced to third parties	
AdvCiv		*BtS*
When two civs make peace and one side pays reparations, then all other civs receive a list of the reparations in the peace-made message.		Third parties only learn that peace was made and, apart from capitulation, nothing about the peace conditions. Can tell though whether a city was ceded – the owner changes, but there is no "captured" message. By monitoring discovered techs and gold available for trade, it's also possible to guess whether a tech or gold was received. No other reparations are possible between AI civs.
Rationale	To make AI behavior more transparent. The peace terms are not so secret anyway (not in the real world either).	
Config	Can be disabled through `ANNOUNCE_REPARATIONS` in `GlobalDefines_advc.xml`.	
Tbd.	Would like to make all trades transparent. See Tbd. at 210 (Civ4lerts) and advc.ctr (city trades). After all, the AI already learns about trades with its worst enemy and all tech trades (for "fear you're becoming too advanced" memory).	

040	AI improvements for settling other continents
See also	[300](#): Barbarian placement on continents without civilizations. [905a](#) gives ships better stats (cost, speed, cargo capacity). [031](#) reduces the impact of distance on AI found value. [113](#) comprises some improvements to Woker ferrying
Tbd.	Would like a Columbian Exchange mechanism that reveals resources in the Old World once a copy is obtained from the New World.

AdvCiv	*K-Mod*
When an AI civ has no city on a continent, it counts unrevealed tiles in the city radius as average tiles when computing found values (with a number of exceptions and restrictions). The Settler may change its mind once it gets to the spot and reveals its surroundings.	Unrevealed tiles don't add anything to found values. This makes the AI very reluctant to settle a continent that it has only explored from the sea.

Rationale	Could instead try to improve AI exploration. Not so easy; how can the AI tell that a continent is large enough for land exploration (without circumnavigating it first)? Should Caravels be routinely accompanied by a Scout/ Explorer?

AdvCiv	*BBAI*
The AI moves Settlers trained in inland cities to coastal cities when there are no city sites on the continent. Fixed a problem with Workers taking up all the space in a ship that is supposed to transport a Settler.	Inland AI Settlers wait indefinitely for a ship. BBAI comment: "*TODO: Go to a good city (like one with a transport) ...*" Update – It seems that ships not having room was the main problem, and that my new `CvUnitAI::AI_moveSettlerToCoast` function is now largely superseded by `AI_load`. Though I'm still seeing the AI use `AI_moveSettlerToCoast` occasionally.

AdvCiv	*BtS*
When a workable tile or an owned tile with a resource is on a landmass where an AI civ doesn't have cities, that civ will (eventually, when Workers don't have much else to do) transport a Worker to the landmass. AI found value computation: As in BtS, but the trade value of offshore resources (i.e. on a landmass without cities) is partially counted.	The never sends a Worker to such a landmass, and tiles there remain unimproved forever. When evaluating city sites, regular yields of land tiles in the city radius that are located on a landmass where the AI doesn't have cities are fully counted, but no trade value and no special yields are counted for resources on such tiles. (See `CvPlayerAI::AI_foundValue` for the meaning of regular yields vs. special yields; complicated.)

See also	[124](#) allows workable resources to be connected without the need for a city or Fort on the same landmass as the resource. This makes offshore resources a little more attractive as the owner doesn't have to choose between connecting the resource (through a Fort) and the full resource yields (through a non-Fort improvement). Change 040 shares some code with [121](#).
Tbd.	`CvCityAI::AI_countNumBonuses` should perhaps count resources on other landmasses in some contexts now that the AI can hook them up.

041	Can train ships in any coastal cities	
AdvCiv		*BtS*
Cities adjacent to any non-lake water area can train ships (except Work Boat – treated as in BtS) and construct a Dry Dock. The AI ignores this change, i.e. builds ships/ Dry Dock only in cities at a size-20 waterbody.		Can train ships and construct Dry Dock only in cities adjacent to a water area of at least 20 tiles. A Work Boat can also be trained when there is at least one water resource in an adjacent water area. Other coastal buildings require only 10 water tiles, which is also the threshold for a salt water lake.
Rationale	Since BtS, it can make sense to train naval units in small water areas because Forts can act as canals into larger water areas. For inland seas and ice-locked water areas, such a Fort is pretty commonly possibe. Implausible that this should not be allowed. Since the AI doesn't have a notion of canals, it needs to stick to the old rules though. This change also eliminates the confusion about two different area size thresholds (10 for buildings, 20 for units). Cities are now either proper coastal cities, or only (fresh water) lake-side cities. Don't want to allow shipbuilding at lakes because it's confusing to see shipbuilding options on the city screen of some inland city with a 1-tile lake. Dawn of Mankind players have also argued that small lakes aren't deep enough for constructing and launching oceangoing ships. Work Boat remains unavailable when there are no resources because, even if there is a canal, players might as well produce their Work Boats in other cities; they're cheap. Don't want players to be confused by Work Boat as the only naval production option in the early game when canals and warships may not be on their minds.	
Tbd.	By allowing Lighthouse in lake-side cities, I could remove the awkward incentive for founding in tiles that are both coastal and lake-side. But I'd rather treat Lake tiles as being a terrain different from Coast and Ocean, and thus unaffected by Lighthouse (and Colossus, Moai); then give Lake +1 food, -1 commerce from the start (i.e. like Oasis but 1 less commerce).	
See also	030 treats ice-locked water as a separate water area and prevents coastal buildings in such cities.	

042	Anticipate border expansion when considering to train Work Boat	
AdvCiv		*BtS*
When deciding how many Work Boats are needed, the AI predicts border expansion 5 turns in advance.		E.g. if the capital has seafood in its outer ring, the AI doesn't consider Work Boat as its very first order. Or in a new city that can expand its borders quickly through the leader trait or Culture process.
Rationale	This was a somewhat disproportionate amount of work to implement. Though I do need it for Japan in the EuroWorld scenario.	

043	Dan Quayle scale adjusted	
AdvCiv		*BtS*
For a "Julius Caesar" victory, a normalized score		15 000 suffice for the best victory, the next

of at least 100 000 is needed; the next best victory is at 75 000, then 50 000, 40 000 ... 1000 is needed to avoid the worst title (Dan Quayle).	thresholds are 14 000, 13 000, ... Dan Quayle is below 3000.
Rationale	Almost all my victories have been "Julius Caesar", so this is apparently too easy to reach for an experienced player. More of a logarithmic scale now, covering a wider interval of scores. It's possible that the original scale worked OK in Vanilla Civ 4; hasn't been changed in either of the expansions.
Tbd.	The score formula needs work too; increase the weight of victory date and difficulty. The main incentive should be to win as early as possible on the highest possible difficulty. Though the bigger issue is that yields increase too much in the Industrial era, so a lot of points can scored by drawing the game out. Should also move the formula into the DLL so that R&F can access it (see comment in `RFTotalScore::updateString`).
See also	707: The Rise & Fall scores are normalized to match this new scale.

044	Changes to the timing of autosaves
See also	106l: No message when autosaving.
	700: The R&F game option changes the timing a bit.
	I've added some initialization steps after loading a savegame to a new function named `CvGame::onAllGameDataRead`.

AdvCiv	BtS
No autosave right after loading a savegame. There's still an autosave after regenerating the map, which replaces the initial auto-save and deletes all other autosaves.	When the initial autosave is loaded, the game immediately autosaves again, overwriting the savegame just loaded and deleting all other auto-saves.

Rationale	It's unnecessary to save right after loading, and the deletion of other auto-saves is somewhat unexpected; should only happen when a new game is started (via the main menu).

AdvCiv	BtS
Per-turn autosaves are created at the start of the active player's turn. Specifically, the savegame is created before diplo popups and votes and after displaying on-screen messages and non-diplo popups. Therefore, after reloading, diplo popups and votes will be shown again (as in BtS), but messages and e.g. the new-era splash screen won't reappear. Exception: When playing multiplayer with simultaneous turns, autosaves happen at the start of a game turn as in BtS.	At the start of the game turn, i.e. right before the start of the turn of player 0.

Rationale	Not sure if the exact timing is ideal with regard to messages and popups.
	Saving at the start of the active player's turn is much better in scenarios, when the human player isn't necessarily in slot 0.
	With simultaneous turns, the order of the player turns is randomized and not stored in savegames, so saving at the start of a player turn could lead to other players missing (parts of?) their turns.
See also	127 makes AI Auto Play end at the start of the human turn instead of the start of the game turn. It's important for testing (reproducibility of errors) that the autosave happens at about the same time as AI Auto Play ends; otherwise, AI Auto Play for several consecutive turns will have a different result than AI Auto Play for one turn at a time. (E.g. if Auto Play ends before the start of the human turn, then the AI will never assign any promotions in `AI_doTurnUnitsPost` when running one turn at a time.)

045	Hide rival buildings

AdvCiv	BtS
In cities where the player's team doesn't have a unit and that he/she can't investigate, only wonders (small and great), defensive buildings and buildings that can be inferred from their plot yield change (Lighthouse, Levee) are shown as 3D models on the main interface. Whether a city is in the fog of war (still) doesn't matter.	All buildings are shown in all revealed cities.

Rationale	Too much information. If a player has all cities of a rival revealed, he/she could check every turn whether a building was completed, and infer that units or wonders are being produced when no (ordinary) building gets completed for some time. With detailed knowledge about the AI code, it might also be possible to deduce the AI strategies and war plans. No one does this, but the game still shouldn't reward it.
	The change could improve performance, but probably just reduces the main memory usage, which isn't an issue in AdvCiv.
	And it makes the important buildings (wonders, Walls, Castles) easier to spot.
	Would be more realistic to show the buildings whenever a city is visible (i.e. not fogged), but hiding them seems better for gameplay; I think showing them only when a unit is inside is a good compromise.
	I'm not updating city layouts when entering or exiting WorldBuilder or Debug mode because doing so seems to (hard to say …) prolong the already quite noticeable delay (at least with debug builds).
Config	XML switch `TREAT_REVEALED_BUILDINGS_AS_VISIBLE` in `GlobalDefines_advc.xml`.
Credits	From Rise of Mankind, though they don't check if the city could be investigated or if there's a unit, and don't show Lighthouse and Levee. Link to source code (1 MB file): [Sourceforge.net](#)
Tbd.	K-Mod comment in `CvUnitAI::AI_nukeValue`: "*It is possible to see which buildings the city has by looking at the map. This is not secret information.*" No longer true, so `AI_nukeValue` should instead estimate the value of the city's buildings.
	Fog of war should apply to the shown buildings, i.e. only those buildings should be shown that were present when the city was last visible. Doesn't seem that hard to implement. Since only a handful of buildings are ever shown in the fog of war now, the additional memory usage should be negligible. On a related note, fog of war for terrain features would also be desirable. Can currently spot rival cities in the fog of war by paying attention to disappearing forests.
	A Civ4lert about completed buildings. Now that the player doesn't learn about most buildings, this wouldn't necessarily generate too many messages. See also Tbd. under [210](#).
See also	[120d](#) removes the Sabotage Building cost from the Espionage screen; could otherwise use that cost to learn about the cheapest building in a city.

046	Pickup of stranded units	
AdvCiv		*K-Mod*
Just some tentative minor improvements to the transportation AI so far.		Units can get stranded on landmasses without friendly cities by razing the only cities or by making peace without conquering any. I think the BtS AI just leaves them there. BBAI has added AI routines for picking up such units, though, judging by the K-Mod changelog, the BBAI code worked only in certain cases.
Tbd.	Not much; don't want to spend time on low-level AI functions.	
	[UWAI](#) could try to predict stranded units when evaluating the tactical situation: Count land units in land areas with an enemy city and no friendly city. Could add these (with an upper bound) to the entangled units or so. Get the relevant areas by going through	

	the enemy cities – iterating through all areas is often costlier because of islands and shelf ice.
Credits	CFC user vedg reported units not getting picked up in K-Mod 1.46 and provided savegames.
See also	040 and 082 also deal with the transportation AI.

When deciding where to place a bumped land unit, landmasses where the unit owner has a city are preferred.	When a unit needs to be removed from the territory of a rival because of an ended war or canceled Open Borders agreement, that unit gets "bumped" to the nearest tile it can legally occupy. Tiles on the unit's current landmass are preferred (tiles on a different landmass are effectively considered to be 1.5 times more distant than their air-line distance).
Rationale	Might make stranded units a bit less common, especially on tiny islands. I don't want to enforce teleportation to a friendly city because this could be exploited for moving units quickly between theaters of war. At least for now, distance should remain the primary criterion.
See also	034 (disabled) was an attempt to avoid bumping altogether. 163 spends the movement points of bumped units.

Added a clause to discourage teleportation of ships into lakes.	
Rationale	If the lake isn't connected to a sea, the ships could be irretrievably lost. I've never had this problem, but this bug report for the "We the People" mod suggests to me that it could happen.

047	Help text for resource tiles revised
AdvCiv	*K-Mod*
The help text shown when hovering over a tile with a resource is now hopefully clearer and more concise. There was also a minor Vanilla Civ 4 bug (marked with id `001` in my code) that had been giving away rival improvements in the fog of war. In the example on the right, the text now simply says: "Grassland, 2 food, 2 production, 3 commerce Uranium (with Fission) Mine Road" If there is no Mine: "Grassland, 2 food Uranium (with Fission, Mine) +2 production, +3 commerce (with Mine)"	karadoc already made some improvements; a comment says: "*I've rearranged and edited some of the code in this section to fix some bugs.*" E.g. says for a Uranium Mine owned by the player when the player has Physics but not Fission: "Grassland, 2 food, 2 production, 3 commerce Uranium, Research: Fission, Requires: Mine +2 production, +3 commerce (with Mine) Mine Road"
Rationale	The example is what prompted my changes.
See also	Somewhat dependent on 135c, which refactors `CvGameTextMgr::setPlotHelp`.

048	Changes to combat odds text	
AdvCiv		*K-Mod*
When more than one unit is selected by the attacker, the combat odds help text shows which of the selected units is about to attack.		When more than one unit is selected, the next attacking unit is chosen for the human player by the AI. Which unit is chosen can only be inferred from the strength value and odds breakdown. In BtS, the same goes for the defending unit. K-Mod indicates the defending unit through the 3D model shown on the map, but, on the attacker's tile, 3D models of multiple units are shown when multiple units are selected.
Combat modifiers that get applied to the attacking unit (i.e. generic combat modifiers) are shown under the name of that unit along with the atttcker's first strikes. Then info about the defending unit is shown, then the combat modifiers that get applied to the defender (i.e. all non-generic combat modifiers) and the defender's first strikes.		
(The shown unit info is a one-line short description with promotion icons.)		When ACO is disabled, combat modifiers that are tied to an ability of the defender are shown first and always in red; then modifiers for river crossing and attack from cargo (which are tied to the attacker insofar that the attacker could eliminate those modifiers through the Amphibious promotion) are shown in red; then the remaining modifiers tied to the attacker are shown in green. The signs are as stated in the unit abilities. ACO instead shows the sign that gets applied in the firepower calculation, i.e. it flips the sign for non-generic modifiers of the attacker because they actually get applied to the defender.
The sign of the combat modifier indicates whether it gets subtracted or added and the color (red or green) whether it benefits the attacker. This applies regardless of the ACO option.		
If ACO is enabled, both the attacking and defending unit are shown by default. ACO is still disabled by default (as in K-Mod).		By default, ACO shows the defending unit as a one-line short description with promotions. The attacking unit is not shown by default.
No separate info about hitpoints.		If either of the two combatants is damaged, the current hitpoints are shown after the combat odds.
Various changes to the arrangement of the ACO menu, the menu text, default settings and in-game text are also tagged with id 048.		The settings "always", "never", "normal" and "alternate" are a bit difficult to understand in the original menu.
Tbd.	When a stack attacks, the map should show only the 3D model of the unit that is about to attack. It's easy enough to set the "center unit" of the attackers' tile accordingly in `CvGameTextMgr::setCombatPlotHelp`, but the game ignores the center unit and shows a mix of 3D models instead. Not sure if and how this can be changed. In any case, the one-line descriptions are still going to be helpful: can't tell from the 3D model which promotions a unit has.	
	Moreover, the 3D model on the info pane should show the unit that is about to attack. Currently `CvMainInterface.py` shows the `HeadSelectedUnit`.	
Rationale	The UI needs to make clear which units are about to fight. ACO already had an option for this, but I don't want to use ACO because I find it too bulky.	
	Hitpoints: The one-line descriptions already include the current strength and the total strength, which is how the UI usually represents hitpoints.	
	Color-coding and signs: Both BtS and ACO assume that each combat modifier can only have one sign. K-Mod's Disorganized promotion breaks this assumption and mod-mods based on AdvCiv may want to break it in additional places. I'm not sure if the BtS approach of showing the sign as stated in the ability (thereby hiding the somewhat countintuitive combat rule about non-generic modifiers always applying to	

	the defender) could be salvaged. I think it's better to tell it as it is and maintaining two modes of displaying the modifiers is extra work.
See also	I've implemented most of these changes also in a mod-mod of Rise of Mankind – A New Dawn 2: [Git commit](#)
Tbd.	This [Git commit](#) could be ported to AdvCiv. Might be nice for mod-mods that use negative modifiers a lot; not sure. Complicates the program logic a bit (and slows it down, but that probably doesn't matter at all).

AdvCiv	BtS
An addendum to the group attack display changes above: Can hold down the Alt key to view the unit with the highest survival odds or, if held while ordering the attack, to force that unit to attack. If the Alt key is not held, the game is free to make a smarter attack. Either way, the odds display shows which unit is about to attack.	When an attack is ordered with multiple units selected, the game always makes a "smart" choice that also takes into account a) how valuable the potential attackers are and b) whether they inflict collateral damage. a) was added by the BtS expansion and it seems that they forgot to update the odds display, which is only based on b), meaning that the displayed odds often don't match what happens when an attack is ordered.
Config	If the smart choice does not have the highest survival odds, a hint about pressing Alt is shown. This can be disabled through `GlobalDefines_advc.xml`. If ACO is enabled ("Odds" tab of the BUG menu), then disabling the "Instructions" option will also disable the hint about the Alt key.
Rationale	At first I just wanted to fix the inconsistency in BtS by showing only the smart choice, but then I figured that there also needs to be a way to just see the best odds.
Tbd.	If I can manage to show the proper 3D models (see *Tbd.* above), there will also be the issue of updating the shown model whenever the Alt key is pressed and released. Not sure if that's doable. Perhaps the smart choice should prioritize high survival odds a bit more; currently it's the exact same procedure as for AI group attacks. `See iOddsThresh` in `CvSelectionGroupAI::AI_getBestGroupAttacker`.
See also	The Alt key functionality depends on [011b](#) (Ctrl key for pre-building).
048b	Replaced a few uses of the word "retreat" with "withdraw" in help text.
Rationale	Game terms should be used consistently. Plus I may want to use the term "retreat" in the context of a [combat round limit](#) in the future.

advc.pf	Changes to pathfinder (previously advc.049)
See also	[104b](#) uses the new TeamPathFinder class for measuring path lengths between cities. [082](#) lets AI units avoids entering enemy borders earlier than necessary when about to start a war. [001i](#) makes the pathfinder respect fog of war on routes. [003s](#) speeds up pathfinding code by precomputing tile adjacency lists. [035](#) (disabled) let's the AI take into account which tiles will flip upon declaring war.
Refactoring changes, performance optimization:	
Moved pathfinding code that cares about (specific) selection groups and units into a separate class	

GroupPathFinder, derived from the KmodPathFinder class.

Derived a class TeamPathFinder from KmodPathFinder for computing paths for "typical" (hypothetical) units of a team. BBAI had implemented largely the same functionality on top of the FAStar pathfinder in the EXE (e.g. in `CvPlot::calculatePathDistanceToPlot`). TeamPathFinder replaces that (much slower) BBAI code entirely. `CvMap::calculatePathDistance` remains in use. That BtS function treats only land/ water and impassable tiles as obstacles, whereas TeamPathFinder also takes into account foreign borders and can distinguish between shallow water (Coast) and deep water (Ocean) movement. TeamPathFinder can replace `calculate PathDistance`, but the latter is more convenient to use, so I'm keeping it in places where performance isn't a concern.

Simplified the logic of karadoc's A* implementation a bit. The original implementation (for unit groups) is still in the codebase for testing purposes. Also changed the public interface a bit (mostly in pedantic ways, e.g. capitalization).

Some tweaks to memory allocation in KmodPathFinder and CvSelectionGroup (function `getClearPathFinder`).

Use a more compact data structure for the nodes in KmodPathFinder. K-Mod had used the same data structure as the FAStar pathfinder.

Created an enum type for the group movement flags in `CvDefines.h`.

Moved the BtS pathfinding functions that don't deal with unit movement from CvGameCoreUtils into a new header `FAStarFunc.h`.

Cached the presence of isthmuses on the map. The check for isthmuses blocking sea movement is the only check that the pathfinders have to make on a per-edge basis (as opposed to per node).

I've included a guess at that memory layout of the FAStar class in `CvDLLFAStarIFaceBase.h`. In the future, this could perhaps be used to replace the results of the FAStar instance that the EXE uses for displaying waypoints with paths computed by GroupPathFinder. Currently, to keep the waypoints consistent with the actual paths taken, GroupPathFinder has to compute the exact same paths as FAStar for human units. This isn't really a problem because we simply want a minimal-cost path, and that's what FAStar computes. Paths of equal cost can be a problem, but aren't currently. [Related CFC discussion](#)

AdvCiv	BtS/ K-Mod
Made sure(?) that, when selecting between paths of equal movement costs for a human player, a path that enters a smaller number of tiles is preferred over a path that enters an extra tile and has a smaller ratio of diagonal moves.	For human player, straight moves are preferred over diagonal moves for aethetic reasons. Taken together with a K-Mod tie-breaker that is supposed to prevent two paths from having the exact same cost, the preference for entering fewer tiles can be outweighed. This results in paths with counterintuitive detours.
Credits	carp provided a savegame ([CFC post](#)).
See also	Earlier version of the K-Mod symmetry-breaking code: [Git commit](#)
Tbd.	I've had to scale some of the decision weights up. Hopefully this has no unforeseen consequences.
As another tiebreaker for paths of equal movement cost, paths that end on a route are slightly preferred.	Routes (greatly) affect movement costs, but only when the move starts *and* ends on a route.
Rationale	Regardless of whether the current path can take advantage of a route, landing on a route has a potential of speeding up future moves. Moreover, a preference for routes should make a rare pathfinder bug involving danger avoidance even more rare (might

	even fix it, indirectly).
See also	Detailed [Git commit](#) message; comments above `GroupStepMetric::canStepThrough` and in `KmodPathFinder::processChild` (regarding "dead end" nodes)

AdvCiv	*BtS*
The AI doesn't build roads in human territory. (But may upgrade human roads to railroads.)	If the shortest path between two AI cities passes through human territory and the human and AI player have an Open Borders agreement, the AI may build roads through the human territory.

Rationale	If the human pillages those roads, the AI keeps rebuilding them, which can permanently occupy several AI Workers. Also, ultimately, humans should be in control over the roads in their territory.
Tbd.	Implemented through a new pathfinder flag. I'm not sure that I'm using that flag in all the right places; should take another look some time to make sure I'm not preventing pathfinder instances from being reused, which could hurt performance.

050	Earth scenario

The [EuroWorld](#) scenario is not part of AdvCiv yet, but I've already made some changes in AdvCiv to accommodate that scenario:

[300](#) prevents large deserts from filling up with Lions.
[140](#) reduces the effects of various world-size modifiers. Earth scenarios tend to be crammed despite their great size, so the BtS modifiers don't work well.
[171](#) lets the AI pick religions in chronological order ("Choose Religions" option) when no favorite is available.
[040](#) allows the AI to improve tiles on small islands in the radius of a mainland city (e.g. needed for Sardinia, which is in the city radius of Rome).
[042](#) (city AI anticipates border expansion) is helpful for the Tokugawa AI – perhaps also in other Earth scenarios.

And all the changes with id 05-something.

Designer's notes: In a scenario, when holding down ALT while hovering over a tile with a resource, the game displays a short text that explains why the resource was placed there if the scenario comes with a game text file containing such designer's notes. See `Civ4GameText_Earth18Civs.xml` for an example.

Disabled when in Debug mode (because ALT key has a different function then).

Tbd.	I'm not sure if I'm ever going to use this – would have to be done for most resource locations (if not all) to make sense. If I do it, I'd let the Java program that I use for generating scenario file also generate the game text file.
See also	ALT key shows combat odds instead of plot help when hovering over a unit. [089](#) disables this when the unit is owned by the player – one can't attack one's own units.

051	Initial game state in scenarios
AdvCiv	*BtS*

In scenarios, civs receive free tech from the game difficulty in addition to any free techs defined by the scenario.	Only free units are received based on the game difficulty (but only if the scenario doesn't define any free cities or units; no change).
Rationale	Don't want the game difficulty to have a smaller impact in scenarios like Earth18Civs than in regular games. Also, the absence of free Archers leads to problems with very early AI-on-AI wars in crowded scenarios.
See also	[CFC post](#) demonstrating a Warrior rush in a Deity scenario. [104u](#) fixes issues with AI initialization in scenarios. [126](#) grants free tech from game difficulty when starting in a later era.
If a scenario places a wonder or a (super) specialist in a Barbarian city, the GP rate of the city remains 0. E.g. the Earth1000AD scenario places Chichen Itza in a Barbarian city. Probably also an issue when Barbarians conquer a wonder.	Specialists and wonders in Barbarian cities increase the city's (generic) base GP rate but not the rate toward any particular Great Person because Barbarians don't have GP units. I suppose this means that GP are accumulated, but no Barbarian GP are born(?).
Rationale	The GP rates should be consistent, i.e. the base rate should always equal the sum of the rates of the individual GP types.
In scenarios, the start era is set to the mean (rounded down) of the player eras.	I don't know if there's a way to specificy the start era in a scenario file. Some scenarios have a `StartingEra` setting in the player sections (after `BeginPlayer`), which perhaps (not tested) gives that player all techs up to that era, but setting a `StartingEra` in the game section (after `BeginGame`) has no effect. The start era is relevant for some modifiers (e.g. cities grow a bit faster when starting Renaissance or later) and for the available wonders.
Rationale	This blocks some Ancient wonders in the Earth1000AD scenario. [This](#) CFC post lists some problematic wonders.
See also	[008a](#) changes the start era threshold for a couple of wonders.

052	Adjustments to AI found behavior for scenarios
See also	Several of the changes under [031](#) were made with scenarios in mind, but also have a significant effect on randomized maps.

AdvCiv	*K-Mod*
When the AI chooses its city sites, found values are randomly increased or decreased by up to 1.5%, and when deciding which site to settle next (Settler Unit AI), the respective found values are randomly increased or decreased by up to 4%. (These random adjustments do not change from turn to turn, but stay the same for each tile for the entire game.) Only applies to scenarios.	Neither AI found values nor the choice of the next site are randomized. Randomness in other choices, e.g. research, scouting, city production and Worker builds, can affect AI found behavior, but the placement of the first couple of cities is often perfectly predictable.
Rationale	Should improve replayability of scenarios. Not much randomness in the site selection because this could move a city from a locally optimal position into one that is obviously inferior, which makes the AI look bad.

The AI considers training a Settler in a size-2 capital when that capital is slow to grow. But not until the capital has a sufficient number of defenders (typically 2). So long as an AI civ hasn't met any human civs, it may leave a single defender in its capital and use the other to accompany the Settler that founds the second city.	The AI normally lets its capital grow to size 3 before training a Settler. Exception: When the capital isn't growing at all or has no decent unworked tiles left. Typically, the AI has enough defenders when it reaches size 3. I'm not sure if it also usually has an extra combat unit ready to accompany a Settler.

Rationale	Capitals with low food but high production shouldn't wait for size 3. (On random maps, such capitals don't really occur.)
	This change is also relevant for non-scenario maps, but especially for the EuroWorld scenario which has unusually dry terrain around some capitals.

053	No Tundra Farms
Tbd.	Want to allow Farms only on Grassland, Plains and Floodplains. Not implemented yet, but I'm letting the AI already treat Tundra river tiles with caution. When the change is made, Tundra river without Forest should also be counted as a "bad" tile.

AdvCiv	*BtS*
AI found value counts only a little extra utility for an adjacent river when a tile has Tundra terrain.	Tundra river tiles are treated the same as other river tiles; quite a bit of utility is added for the river.

Rationale	Earth scenarios have large taiga areas along rivers, and these areas are not supposed to be deforested and farmed. Taiga soils are very badly suited for agriculture (and actual tundra soils are even worse). On randomized maps, this shouldn't make a big difference as they generate few Tundra rivers. Settling the far north can still be worthwhile when there are bonus resources.

054	Visibility of game options
See also	[250b](#): Advanced Start and SPaH

AdvCiv	*BtS*
The "Permanent War or Peace" game option is shown on the Custom Scenario screen but not on the Custom Game screen.	The option is visible on both screens, but it's equivalent to "Always Peace" unless a scenario sets certain civs to be at war from the beginning.

Rationale	The option is confusing and superfluous in non-scenario games.
Config	Set to invisible in `Civ4GameOptionInfos.xml`. Can be toggled to make the option visible on the Custom Game screen too (as in BtS). The AdvCiv DLL makes the option visible on the Custom Scenario screen regardless of the XML setting.
Tbd.	Would be nicer to add a field to `Civ4GameOptionInfos.xml` that determines whether an option shows up on Custom Scenario.

"Lock Modified Assets" is never shown on the (multiplayer) Staging Room screen.	Shown both in singleplayer and multiplayer but has no effect in multiplayer. (`CvGame::init` disables the option, ignoring the setting on the Staging Room screen.)

Rationale	Not sure why the original developers disabled this (it's already that way in the Vanilla

	Civ 4 code). Perhaps modified assets will result in OOS errors anyway.
Tbd.	Should temporarily disable the code in `CvGame::init` and give this a try. If the option actually works, it could be nice to have for some players.
"New Random Seed" on reload is never shown on the Staging Room screen.	The option is ignored in multiplayer. No new seed is generated (confirmed here by Afforess). That said, with simultaneous turns, `CvGame::updateMoves` constantly uses numbers from the synchronized RNG for randomizing the turn order, and this makes random outcomes non-reproducible.
Tbd.	If the EXE reports the Staging Room setting to the DLL (I haven't checked), it will be easy enough to the `isNetworkMultiPlayer` check from `CvGame::read`. Could then re-enable the option (though it'll still be meaningless with simultaneous turns).
	Can something be done to make randomness reproducible with simultaneous turns? Can't really debug with this option currently. Mongoose Mod might fix it, but is closed-source (change log, 20 Dec 2010: "*devised a simple, effective way to lock random seeds in Network Multiplayer games when the NewRandomSeedOnReload option is not enabled*").

055	Global Warming (GW)
Tbd.	See WIP section
See also	Melting ice isn't really supported by 030 (q.v.).
	002l prevents the GW sound from playing more than once per turn.
	111 makes it easier to pillage friendly routes (and thus easier to disconnect Coal and Oil).

AdvCiv	*K-Mod*
When GW hits a Plains Forest or Jungle, the feature is removed and the terrain remains unchanged. Vegetation on other terrain types is unaffected by GW (but the underlying terrain can change).	GW events always turn Grassland Forest and Plains Forest into Jungle, and remove Grassland Jungle and Plains Jungle. This behavior may have been adopted from the Global Warming Mod.

Rationale	Forest to Jungle seems quite unrealistic to me. The bad health makes sense (think of the spread of the tiger mosquito), but vegetation becoming denser or wetter is too strange.
	Removing vegetation is generally problematic because late-game maps tend to be mostly cleared before GW even begins.
Config	Can set `PROTECT_FEATURE_ON_NON_DRY_TERRAIN` in `GlobalDefinesAlt.xml` to 0 in order to allow GW to remove vegetation from Grassland as well. That should probably be done when giving Forest Preserve the ability to protect vegetation (see below).
See also	CFC post where I weigh the change described above against the alternative change described below.

(Disabled again in favor of the change described above.)	GW doesn't directly destroy Forests, but it can turn them into Jungles and can destroy Jungles. If that happens to a Preserved Jungle, then the improvement is destroyed (as it requires
Preserved Forests and Jungles can't be removed by GW. The terrain type can change, but not to	

Desert. If a Plains Forest Preserve is randomly selected for a GW event (which normally removes the Forest and, failing that, turns the terrain to Desert), then no GW event takes place instead – i.e. the target tile isn't re-rolled.	vegetation).

Credits	Proposed by Cruiser76 ([post](#))
Rationale	Since the goal is to buff Forest Preserves (or at least not to hurt them with the Forest removal change), I didn't want to block terrain changes – those will often change Tundra to Grassland. Protecting the Forest Preserve improvement itself (which requires a feature) might be more elegant, but it's difficult to ensure that a GW event won't destroy an improvement (in general; not for the Forest Preserve in particular).
Config	New optional XML tag for improvements: GWFeatureProtection If a (positive) value less than 100 is used, then the feature is protected only probabilistically.

Reduced the per-turn probability of GW events to 3/5 of its value in K-Mod.
Upd.: Reduced it again in AdvCiv 1.03, this time to 1/2 of the K-Mod value. And, in AdvCiv 1.04, introduced a multiplier based on the initial team count to decrease the value (further) on maps with fewer than 8 initial teams and to increase it on maps with more than 8 initial teams.

Config	Through `GlobalDefinesAlt.xml` (but the team count adjustment is hardcoded in the DLL).
Rationale	Player feedback suggests that GW was starting unexpectedly early. The warning about the sustainability threshold still comes as early as before, but isn't as much of a concern as the GW events, which alert the player turn after turn. crullerdonut's play reports also say that GW anger wasn't much of an issue. So, as a stopgap measure, let's deal with the event probability. I suspect that the issue is mainly that tech costs are higher in AdvCiv than in K-Mod – at least on the higher difficulty levels – leading to longer games and (far) more GW events overall. *Upd.* (v1.03): Game length still appears to the main factor. Sometimes, games will descend into near-permanent warfare when Riflemen become available; those games can approach the time limit. Don't want such games to desertify much of the land area most of the time. *Upd.* (v1.04): All-AI games on Huge maps with about 18 civs seem to be considerably less likely to be severely affected by GW than games on smaller maps with 8 and fewer civs. I'm guessing that this is because games with a higher number of factions tend to produce more heterogeneous tech progress. When some civs don't manage to fully industrialize before the game ends (and don't get conquered by more advanced civs either), then there is a lot less GW.
See also	Posts linked in [WIP](#) section. [K-Mod Git commit](#) that tweaks the same variable that I have tweaked. The commit message suggests that karadoc had also concluded that the probability needs to be adjusted to the typical game length. [251](#), [910](#): Tech cost adjustments.

Tweaked a parameter to increase the GW "severity" rating shown on the Environment tab. The rating is a visual clue about the overall impact of GW, but also affects anger from GW.

Rationale	Not sure if the anger needs to be higher. I mainly felt that – despite the decreased event chance in AdvCiv (which does not enter into the severity rating) – the rating tended to understate the severity as I perceived it. E.g. one event per turn hitting a

	land tile owned by the human player is already pretty severe in my book.	

AdvCiv	K-Mod
GW turns Tundra into Plains. Plains near cold terrain (coldness score based on adjacent Tundra, Snow, Ice) can't be turned into Desert. (But the cold Terrain could eventually turn into Plains, so everything can still get desertified eventually.)	Into Grassland, referred to as "temperate" terrain in the code, whereas Plains is referred to as "dry" terrain.

Rationale	Grassland is too big an upgrade. I don't agree that Grassland should be interpreted as (always) being wetter than Plains (which is apparently the idea in BtS too – so I don't blame K-Mod). It's more helpful to interpret the terrain types in terms of their yields – because the yields matter for gameplay.
	Don't want Tundra to turn into Desert in two steps though; hence the special treatment for Plains near cold terrain.
	A somewhat undesirable side-effect of this change is that Tundra Forest can lose its Forest after two GW events instead of 3. Well, I don't think double events are all that relevant for the overall deforestation rate (not until GW becomes really bad) and it's not unrealistic for taiga (Tundra Forest) to be vulnerable to forest fires as global temperatures increase.

AdvCiv	BtS
Announcements of GW events that happen near a city of the active player's team say how the tile has changed (terrain, feature, improvement).	The announcement only says that "Global Warming strikes near" the city; clicking on the announcement fouces the camera on the tile.

Rationale	Players can figure out how the tile has changed, but if the message already says that e.g. Grassland has changed to Plains (a common event), then players don't even need to look at the tile – probably doesn't require any attention.

AdvCiv	BtS
A GW event near a rival city is announced as described above if only one rival city is affected by GW on the current turn; otherwise, a single announcement lists all affected rival cities (without saying which specific tiles were affected and how).	Only events near cities of the active player are announced. (With K-Mod, the expected total number of events per turn is shown on the Environment tab of the Economic Advisor; so a player can guess how many unannounced events are happening.)
Melted water Ice on unowned tiles that aren't within or adjacent to a city cross are announced in a single message that only says how many tiles were affected on the current turn.	

Rationale	Not terribly relevant, but players don't like missing out on changes to the known game state, and the announcements give the player a better impression of how much GW is affecting the world and that rivals are equally (proportional to their land area) affected.
See also	Screenshot attached to this CFC post (1st attachment).

056	WBSave format

AdvCiv	BtS
Can read scenarios so long as the player ids in the WorldBuilder save are less than the maximal number of civs (`MAX_CIV_PLAYERS`) allowed by	If `MAX_CIV_PLAYERS` is changed in the DLL, then WorldBuilder saves created with a different `MAX_CIV_PLAYERS` value become unreadable. In

the current GameCore DLL.	particular, mods that allow 48 civs can't load any of the official scenarios. (The scenario files can be converted fairly easily though by inserting 30 team and player sections for 30 dummy civs into the scenario file.)
Config	The WB reader/ writer is implemented in `Python\pyWB\CvWBDesc.py`. The maximal number of civs can only be changed in the DLL (`CvEnums.h`).

057	Changes to impassable terrain (for mod-mods)	
AdvCiv		*BtS*
When a terrain or feature type is impassable for a unit, that restriction doesn't apply to tiles owned by the unit's owner.		Owned impassable tiles can only be entered by sea units; e.g. Work Boat can enter owned Ocean tiles. No exceptions for units of other domains.
Rationale	A comment in the code makes clear that the special treatment of sea units is deliberate, but I guess the developers just weren't sure how the terrain-/ feature-impassable abilities would be used by modders and felt that exceptions would only be confusing.	
	Generally, I feel that indigenous units should be able to handle their native countryside and that any workable tiles should also be passable. It could make sense e.g. to prevent Artillery from entering Jungle under any circumstances, but the terrain-/ feature impassable abilities allow entrance with a particular technology, so they're not quite the right tool for such a strict requirement in any case.	
	As for the rule that allows trade on owned water tiles – there's the `TerrainTrades` tag for that (`Civ4TechInfos.xml`).	
See also	[CFC post](#) by Cruiser76	
When hovering to move into terrain that one of the selected units is unable to enter, help text says how this could be amended: If the tile can be owned (not too far off the coast) and, help text says that owning the tile will make it passable. If a tech will allow the unit to enter, then help text names that tech.		Help text never explains why a move isn't allowed. (Help text for Peaks and Ice says "IMPASSABLE" at all times.) Tech allowing a unit to enter otherwise impassable terrain is shown in the unit's Civilopedia article and when hovering over the unit icon in the tech tree.
Config	By default, tech that is more than one era ahead of the active player's current era isn't shown. This can be changed through `SHOW_IMPASSABLE_TECH_ERA_DIFFERENCE` (`GlobalDefines_advc.xml`).	
Rationale	For mod-mods; specifically, it was requested [here](#) on CFC.	
	In AdvCiv, the tech info is only relevant for Work Boat. The ownership info is probably superfluous (I also haven't spelled out that team ownership is sufficient – for lack of space), but showing a tech requirement without mentioning the ownership rule might be a bit misleading.	
See also	[089](#) also explains in help text why a unit can't enter a tile (defenders maximally damaged).	
The AI uses a fingerprint (`CvPlayerAI::AI_unitImpassables`) to check whether units have the same impassable terrains and features.		A mere count is used (`AI_unitImpassableCount`).

The most significant bits of that fingerprint also contain a count of the impassable terrains and features so that ordering units by their impassable fingerprints orders them (primarily) by their impassable counts. 0 still means that there are no impassable terrains and features.	

Rationale	Obviously not ideal to treat e.g. a unit that can't enter Desert and one that can't enter Snow as compatible wrt. their movement restrictions. And it was easy enough to change and without paying a performance penalty.
See also	The `isAny...Impassable` functions added by 003t are helpful here.

The AI ensures for each unit group that no unit in the group a higher impassable count than the unit leading the group. This is accomplished by disallowing units to join a group if the newcomer would become the group leader and has fewer impassables than the current group leader, or if the newcomer wouldn't replace the current group leader despite having more impassables. (I did not change the algorithm for selecting group leaders.) When a human civ comes under AI control (e.g. AI Auto Play), groups that don't satisfy the invariant stated above are split up. When the AI type of a newcomer and the current group leader are the same, then both need to have the same impassable fingerprint. For the sea assault AI type, this isn't always enforced.	Unit groups (`CvSelectionGroup`) represent AI unit stacks. The behavior of an AI stack is determined by the AI type (`UnitAITypes`) of the unit that leads the group. Group leadership is mainly determined by AI type (cf. `CvUnitAI::AI_groupFirstVal, AI_groupSecondVal`); units with a specific role, e.g. city attackers or settlers, are preferred over units with more general or supporting roles (e.g. reserve, escort). Civs under human control also have groups; those groups represent current and past units selections. (The current selection is also stored by the EXE → `CvDLLInterfaceIFaceBase::getSelectionList`.) AI units that are unable to enter some terrain or feature (positive impassable count) cannot join a group with a different impassable count. For sea assault units, impassable counts have to be equal in any case (through a BBAI change).

Rationale	The BBAI behavior is OK for the BtS units (Galley, Trireme), and I haven't changed anything about that (I hope). Equal impassables are especially important for assault units, but difficult to ensure because sea assault groups are formed in multiple places (whereas AI group formation was otherwise centralized by K-Mod into `CvUnitAI::AI_omniGroup`). BBAI doesn't ensure equal impassables for sea assault groups either; there's special code for splitting out Galleys, so this is probably OK. For impassables added by mod-mods, requiring equal impassable counts seems too strict; can easily break the AI entirely when units aren't allowed to escort settlers. My invariant allows a support unit (low `AI_groupFirstValue`) to join a group with a high `AI_groupFirstValue` (`UNITAI_SETTLE` has the highest one) even if the group leader has stricter movement rules than the newcomer. This seems like a sensible principle, and it should avoid problems with paths computed for the group leader that other units in the group may not be able to follow (not sure if this can actually happen; I guess the pathfinder checks movement rules for each unit in the group anyway). Perhaps most importantly, the invariant makes it sufficient to check the impassables of group leaders in `AI_omniGroup`, i.e. it saves time. The BtS behavior seems inconsistent insofar that it allows units without impassables to join groups with impassables, i.e. it seems that a symmetrical check is missing.
See also	The validation of AI groups after taking over from a human player are implemented through 127, 127c and 700 (`RiseFall::setPlayerControl`). CFC discussion

Tbd.	Currently, a unit with one impassable could still join a settler with two different impassables. Really need a set inclusion check, and not just for the case when group-first values are the same. Don't know how to approach that.
057b	Only relevant for mod-mods: Invisible settler units don't wait for an escort. (With the BtS AI code, they wait in vain forever.)
Rationale	Someone had this problem in their AdvCiv-based mod. The BtS AI code prevents invisible and visible units from forming a group. I don't know if there's an important reason for that other than that invisible units tend to have different tasks than visible units. I'd rather that not mess with that. Letting an invisible settler go it alone isn't ideal because the city will be defenseless at first, but at least the settler gets to take advantage of its 2 movement points this way.

058	Conceal player identities (when appropriate)	
AdvCiv		*K-Mod*
Functions `getKnownName`, `getKnownPlayerColor`, `getKnownCivDescription` and `getKnownCivShortDescription` added to CvPlayer. Those functions take a team as parameter and return placeholder information (e.g. "unknown player") when that team isn't allowed to know the identity of the player. Reverted the K-Mod changes. Calls from the EXE to `getPlayerColor` are redirected to `getKnownPlayerColor`.		Modified `CvPlayer::getName, getPlayerColor, getCivilizationDescription` and `getCivilizationShortDescription` so that they return placeholder information when the active (human) team isn't supposed to know the identity of the player. Perhaps that's why karadoc chose to make the placeholder behavior the default? So that the EXE would display the correct colors. He probably wasn't aware of `.def` files, so my solution wasn't available to him.
See also	The K-Mod approach was essentially unworkable for the R&F component, which preserves interface messages sent to AI players because the human player might later take control.	
Credits	carp made me aware of the problems with the message log. (CFC post)	
Rationale	The situations in which placeholder information should be displayed are really very few. The K-Mod code had lead to placeholder info in some inappropriate places though (e.g. in BBAI logs). Still, it was a good thought to put the code for generating the placeholder info into a single place, so I've kept that part of the K-Mod code.	

059	Help text for city health/ happiness from surrounding tiles	
See also	Based on refactored code for 901, which also changes the rules for health/happiness effects across borders. 004b shows health/ happiness help for a future city when a Settler is selected.	
AdvCiv		*BtS*
When hovering over a tile in the radius of a city owned by the active player or a tile owned by the active player in the radius of any revealed city, health and happiness effects from improvements and features are shown. The hover text says which cities are affected (this can include cities not owned by the active player). Exception:		Tile hover text doesn't show any health or happiness effects from features and improvements (only shows effects of resources). The city screen shows a tally of the health and happiness from surrounding tiles.

Health effects between -0.49 and 0.99, i.e. when there is only an ordinary Forest, Jungle or Flood Plains, are only shown when a Worker is selected in the tile that the mouse hover over.	
Rationale	The +1 happiness from Forest Preserve should be shown in hover text; it's the main reason for building Forest Preserves. I want to show happiness and health from features and improvements together so that city names don't need to be listed multiple times. (Traits can grant happiness from features and, through 901, improvements can grant health; both unused.) However, I don't think I want to show the small fractional health effects of BtS features all the time; they're not so important and players are used to them. Hence the complicated conditions for when to show the new hover text.

The city screen uses slightly more generic explanations for happiness and anger from features.

AdvCiv	*BULL (not in K-Mod)*
When hovering over a Worker action button, the changes in happiness and health for nearby cities are listed. Fractional changes are shown in gray if they'll be lost to rounding.	Optionally shows a hint in the action hover text if the Worker build will decrease health in any nearby city. That hint also says whether that city has greater excess health than excess happiness or vice versa.
Rationale	The comparison between excess health and happiness doesn't seem very helpful and it can't be properly explained with the available space. Also, without that extra info, there's no need to make the new hover text optional.

060	Add-to-spaceship not shown after part completed	
AdvCiv		*BtS*
After completing a Spaceship (SS) part, the player is only taken to the SS screen if the SS is ready to launch.		Except in networked multiplayer, the player is taken to the SS screen after every completed SS part. The SS screen was added in BtS.
Config	BUG menu, "General" tab under "Splash Screens"	
Rationale	Finishing one of the sixteen parts isn't such a grand event.	

061	Help text that lists the units in a tile rewritten	
AdvCiv		*BtS*
The available number of lines is estimated based on the context (whether the mouse is on a tile or on a unit flag or an indicator bubble), screen resolution and font size (if 002b is enabled, size 14 is assumed, otherwise size 12).		Fixed limit of 15 units to be listed on a separate line each.
If a player has more than 1 unit in the tile, that player's units are listed under a heading, e.g.: Tokugawa (2 Units): Archer 3 str. Combat I Archer 2/3 str. If a player has just 1 unit, the owner is shown in-line. Due to the larger font, this line sometimes wraps into a second line.		The owner is always shown in-line: Archer, Tokugawa, 3 str. Combat I Archer, Tokugawa, 2/3 str. No per-owner unit counts are shown anywhere.

The unit count in the heading is broken down into up to three categories if units of more than one category are present. The categories are "army" (military land units), "navy" (military sea units) and "other". E.g.: Tokugawa (1 army, 1 other): Rifleman 14 str. Airship 4 str. In some cases, the heading is omitted, e.g. when all units in the tile are owned by the active player and they're few enough to count at a glance. If there isn't enough space for all the units and headings, the information shown about the tile's center unit is (slightly) shortened or shortened to a single line.	Same, though it can't be shortened to a single line. (The center unit is the one shown on the map as a 3D model.)
If there still isn't enough space, units that have both the same owner and type are aggregated into a single line, e.g.: Archer (2), 2.5/3 str. Combat I x1 This is applied to all units except for the center unit and units with an attached Great General.	If there are more than 15 units, the first 15 are still listed on a separate line each. The rest is aggregated as described on the left (minus the exception for Great Warlords), e.g: Archer (2), Tokugawa, 2.5/3 str. Combat I x1 These lines are often so long that they wrap around.
If the algorithm expects that the aggregated entries will still require more space than is available (this should happen very rarely), a line with a warning is displayed and, for some civs, only the heading with the unit counts is shown.	No such fail-safe mechanism.
The unit owners are ordered as follows: First units without known owner, then those owned by the civ that owns the center unit, then ordered by (total) unit count, then by slot id.	If the active player has units in the tile, then those units are shown first. Then ordered by player slot id (I think).
The units of a given owner are ordered as follows: First transports and units in cargo (interleaved so that one can guess which unit is in which transport). Then units of the same type as the center units, then units of the same domain as the center unit. Then ordered by domain, then by combat strength, then by unit id (or unit type id if aggregated).	Transports/ cargo: same. Then ordered by how recently the unit arrived in the tile (I think) and unit type id.

Config	New option "List Units per Owner" on the "Map" tab of the BUG menu; enabled by default. All the changes above only apply if the option is enabled.
Tbd.	Use the change tag (see `config\Tech Window.xml` for an example) to toggle the Unit layer twice when the status of the new option changes. Currently, the help text attached to the Unit bubbles isn't updated until the Unit layer is manually toggled. The "More Naval AI" mod allows human to cycle the center unit of a tile through key presses ([Git commit](#)). Would be nice to have some means of viewing the full information about any unit in a foreign stack. Not sure if this would be fully compatible with AdvCiv. I think there's some non-UI code by me that accesses the center unit just to get some arbitrary unit in a tile. Shouldn't matter how the center unit was chosen then – but I'm not quite sure. Also a bit laborious to merge.
Rationale	The main issue was that the BtS code wasted too much space. With the larger font

(002b), parts of unit list were frequently unreadable. The wrapped-around lines were also difficult to read. The per-owner headings solve these problems and provide unit counts, which I had wanted to add in some form and place for some time.

Another issue with the BtS algorithm: A mix of aggregated and non-aggregated info is confusing, especially when the units shown separately are chosen mostly arbitrarily (unit type id and slot id should only be used as tiebreakers).

See also	[101](#) and [187](#) add some city-related info (revolts, air unit capacity) to the hover text of city tiles, potentially limiting the vertical space available for units.

The owner of a Privateer is revealed if the Privateer shares a tile with a visible non-Privateer unit or when the Privateer is in a City or Fort.	Only revealed when the Privateer is in a City or Fort.
Rationale	Revealing Privateers in port makes some sense because units that are always hostile (though BtS actually didn't check this) can only enter the port of their owner (well ... or of a teammate). The same reasoning should apply to a Privateer stacked with visible units whose nationality isn't hidden.
	With the changes to help text, an anonymous Privateer stacked with other units would have to be the center unit, and this would make it stand out more than I'd like.
See also	[007](#) reveals the owners of Privateers when in Debug mode.

062	AI diplo comment states the reason for canceling a vassal agreement
AdvCiv	*BtS*
When the AI cancels a vassal agreement, the diplo popup says, in vague terms, why the agreement is being canceled.	Always says "It appears you are too weak to protect us ..."
Rationale	Got the idea from [this](#) thread (K-Mod subforum). In AdvCiv, I don't think an explanation is really necessary, as voluntary vassal agreements to humans are rare and normally canceled because of the vassal's power rating, but it's still nice to have. Also, "too weak to protect us" is misleading when the ratio between vassal power and average power is responsible for cancelation.

063	Changes to BUG's Actual Effects mod component
AdvCiv	*K-Mod*
Show the Actual Effects production, commerce, happiness, health, specialist and maintenance help text when Alt is held down or when the respective options are enabled.	K-Mod already treats the Actual Effects building help text this way, but the other options aren't tied to any modifier key.
Rationale	Perhaps karadoc just didn't find these options helpful at all. They don't seem very useful to me, but there isn't really any downside to my change, and it could convince players who tend to enable all the BUG options to leave Actual Effects disabled, resulting in a less cluttered UI.
	K-Mod doesn't include the food and defense options. I suppose these were deliberately omitted as the code seems very easy to merge. As far as I can tell, these options would only apply to Baray, Walls and Castle, and I don't think that justifies two additional checkboxes on the BUG menu.

Rewrote parts of the explanation text for the Actual Effects options as the original text was very

sparse. E.g. for the production option: "When checked, displays the buildings that will affect the production rate and by how much." Displays it where? Which buildings? The options are now under the heading "Absolute Effects", which seems more descriptive. ("Net Effects" would be another alternative.)

Added a checkbox for the net effects of specialists to the BUG menu (and wrote help text for it).	K-Mod only shows those effects when Alt is held down; can't permanently enable them. (It seems that BULL always shows net effects of specialists; an option exists in the configuration files, but isn't shown anywhere on the BULL menu.)
Rationale	Maybe an oversight in BULL. The options are good for making players aware of all the features of the Actual Effects mod.
Credits	VDNKh made me aware of the missing option ([CFC post](#)).

064	BUG Whip Assist; changes to production overflow.
AdvCiv	*K-Mod*
Merged BULL's `HurryOverflow` option (overflow info in the "whip" button's help text) and enabled it by default. Adapted the computation to AdvCiv's treatment of excess overflow (see below) and revised the help text. "Include Current" extra option also merged, and enabled by default. Changed it so that current overflow and feature production are included in the predicted overflow regardless of the "Include Current" option. The (other) `WhipAssist` option (the option names in the code are a mess; that one shows overflow info directly on the production bars of cities) remains available and disabled, and now uses the same (C++) code as the `HurryOverflow` option, and so do the hurry Civ4lerts. Redundant Python code (`CvMainInterface.py`, `Civ4lerts.py`) commented out.	Of the three BULL options related to hurry overflow (each with an extra option "Include Current"), two require DLL changes, and karadoc didn't merge these. One of these two, `HurryOverflow`, had been shown on the BUG menu but had no effect when enabled. All were disabled by default.

The pure-BUG option (`WhipAssist` - overflow on the production bar) was working correctly. The "Include Current" option adds the city's current production rate to the overflow resulting from hurry production. This is correct insofar that the overflow after pressing hurry is going to be increased by the city's production rate, but the production rate after sacrificing population will be used, and the Whip Assist can't predict (doesn't try to) which citizens are going to be sacrificed and whether citizen automation will cause the remaining citizens to be reassigned. |
| *See also* | 064b (see below) adds similar help text to the production yield breakdown. |
| *Rationale* | While I think that players should use the "No Slavery" option ([912d](#)), the mod should still offer UI support for Slavery, at least until a civic to replace Slavery is implemented.

The last missing Whip Assist option would show hurry info on the city billboard (or it's help text?). I suppose that the fast Deity-level players, who don't like to enter city screens all the time, use this. Could perhaps merge it along with other BULL changes to the billboards, but AdvCiv perhaps just isn't the right mod for such players. |
| *Credits* | As far as I can tell, EmperorFool implemented the option that I've merged from BULL. |
| Separate option on the BUG menu ("City Screen") for disabling hurry tick marks. Disabled hurry tick marks by default. | The "Tick Marks" option on the "General" tab enables all tick marks, including the little yellow ones that show how much production is needed to decrease the population loss from Slavery. |

Rationale	Two different tick marks make the production bar look cluttered, and it's nearly impossible to guess what the hurry tick marks are for.
See also	078: Other changes to the BUG progress bar options.

064b	Changes to overflow production rules. See the bullets in the blue boxes for the actual changes. For context, let's go through the change history first:

Vanilla Civ 4	*Warlords*
Overflow production is capped at the maximum of the total production cost of the completed production order and the city's base production rate (hearsay: this upper limit may have been different prior to patch 1.61); any excess overflow is discarded. As a comment in the code says "*to eliminate prebuild exploits.*" (I.e. to prevent players from producing items that take the city less than a full turn for several turns in a row, stacking up overflow toward e.g. a wonder, spaceship part or military units whose tech requirement is still being researched.)	Excess overflow is turned 1:1 into gold.
Unofficial Patch 0.21 for BtS 3.17 (link)	*BtS patch 3.19*
From patch notes: "*Limited which production modifiers affect gold from production overflow.*" Lets the same modifiers apply as for Wealth.	From patch notes: "*Build-specific production modifiers are no longer counted for overflow gold calculations.*" However, the implementation works quite differently, usually discarding excess overflow without compensation. See e.g. this post for details. This post by one of the authors of the official patch confirms that Firaxis had only meant to clean up the code from the unofficial patch.
Unofficial Patch for BtS 3.19	*K-Mod*
From v1.0 patch notes: *Kept overflow fixes from 3.17 unofficial patch.* From v1.4 patch notes: "*Fixed extra overflow production bug when stopping culture process after border pop.*"	Same as the latest unofficial patch for buildings, but, for units, excess overflow is spent on additional units of the same type if possible. If there is enough overflow, multiple units can be trained in one turn.
Issue with K-Mod rule	• Considering that overflow gold still occurs for buildings (e.g. Walls) and national units, K-Mod adds complexity to rules that were already complicated. • On the highest difficulty settings in the late Classical or early Medieval era, AI civs that see a large enemy stack approach can routinely produce more than one Archer per turn through Slavery. This raises the cost for conquering mature cities considerably – which might be OK for human-AI wars (though YouTuber Lain got very upset about it in this video), but I suspect that it can lead to indecisive AI-AI wars, and I don't want the invading AI to train and bring as many units as it takes. • Usually, when excess overflow occurs, K-Mod produces another unit only partially. To redeem the production spent, the player will have to continue training that unit. But this will result in even more overflow as it's going to be a cheap unit and maximal overflow is already being stored. For example, if a city with 105 production per turn trains a Spy (40 production), it'll store 40 overflow and put 25 into another Spy. If the player finishes that second Spy to avoid missing out on the 25

	production, 130 overflow will result in 40 overflow stored, two more Spies and 10 production for a fifth one. Ultimately, this leaves it up to the player to avoid excess overflow in the first place. • When there are production orders queued up, it's counterintuitive that excess overflow is spent on an additional unit that the player didn't ask for and not on the queued orders.
See also	[001v](#) fixed an exploitable bug in the K-Mod overflow code. The fix was removed along with the K-Mod code.

AdvCiv: Same as the latest Unofficial Patch, i.e. fully reverted the K-Mod change, and made the following tweaks to overflow, chopping and Slavery:

• The overflow cap is set to the maximum of the city's food store capacity and (as as in K-Mod/BtS) the city's base production rate. The production cost of the completed order no longer matters.

Tbd.	Could relax the cap a bit more, e.g. 150% of the food store capacity.
Rationale	The overflow cap should be chosen with two goals in mind: (a) Prevent players from stacking up substantial amounts of production and (b) set the cap so high that it's unlikely to be exceeded unintentionally. As for (b), using the city's base production rate is fair enough. As for (a), however, the cost of the completed order is not a good indicator for problematic stacking. What should matter is the cost of the production order that the stacked production is going to be used for. This is unknown, so some estimate based on the overall game progress should be used. Since overflow is somehow stored by the city, using the food store capacity makes sense to me. Conveniently, the food store capacity is already adjusted to game speed, map size and start era.

• Production from chopping is no longer converted into overflow, and instead remains stored separately until the city needs it.

Rationale	The accumulation of chopping production isn't an issue (or at least not one that overflow conversion can address) because the player can always preserve chopping production for a future production order by simply not chopping the Forests until needed. Granted: It can be more efficient in terms of Worker utilization to chop Forest ahead of time and keep the yield stored at the city; however, the BtS/K-Mod rules already allowed this by having the city build a Process (which can't use up chopping production), and my rule change has the substantial benefit of preventing chopping production from ever getting converted into gold, and preventing hurry production from stacking with chopping production to generate overflow (see below).
See also	Wonder fail gold can still convert chopping production into gold. See *Tbd.* under [123f](#).

• The formula for the population to be sacrificed for Slavery takes into account the current overflow and chopping production: That production is already guaranteed and doesn't need to be covered by Slavery. Consequently, Slavery can't be used at all when the entire production cost is already covered.
• The same goes for the computation of gold for rush buying, however, in this case, the current production rate of the city is fully taken into account.

Rationale	Slavery generates the most overflow when applied to an order that is already about to complete. Ideally, hurry production should only add as much production as is needed to complete the order at the end of the turn, resulting in 0 overflow. Of course production from Slavery comes in chunks of 20 or 30 hammers, but the principle still applies.
	For rush buying, the entire production rate can be taken into account. If the player rearranges the citizens afterwards, there can be overflow (can't be helped) or the production order may no longer complete at the end of the turn. In the latter case, the

| | player can always rush buy again to close the gap. |
| | As Slavery removes at least one citizen, the city's regular production from tiles can't be counted on, but the currently stored overflow and chopping production are entirely reliable. Strictly speaking, 1 production from the city tile is also guaranteed and my original implementation had counted that 1 production. I've reverted that (see `CvCity::minPlotProduction`) because reducing overflow by 1 isn't worth the additional complexity. Related post of CFC: link |

AdvCiv	*BtS/K-Mod*
When an order is about to complete, the production yield tooltip on the city screen says how much overflow and chopping production will carry over to the next order, and how much gold, if any, will be generated.	BULL (see 064 above) shows info about overflow production (including chopping overflow, which can't carry over separately) and gold, but only when Slavery is involved.
If overflow is converted into gold, an on-screen message says how much.	The overflow production is converted silently. (And K-Mod's additional units also appear without any notification.)
Tbd.	Would be nice to show a breakdown of hurry modifiers in the button tooltips. Also, the impact on stored chopping production isn't shown on the Slavery button, however, Slavery I'm hoping to remove eventually anyway, whereas rush buying is here to stay.

064c	Changes to hurrying wonders
AdvCiv	*BtS*
Can sacrifice at most 3 population at once (perhaps especially relevant for wonders, but applies to all types of production).	Can sacrifice up to half a city's population rounded down.
Rationale	To ensure that wonders can't be constructed abruptly, but mostly because the sudden high population loss is jarring.
Tbd.	I don't like the high hurry penalties for great wonders as I don't see a problem with wonders getting hurried so long as they're not completed in just two turns. Also, the penalties can be bypassed through overflow (although 064b makes this more difficult). And there should be a single penalty for all great wonders, not 100% to 300% as is currently the case.
	I've tried setting the penalty to just 50% (i.e. 20 production from Slavery instead of 30, same as for national wonders), but, for the time being, this would exacerbate problems with fail gold (see *Tbd.* under 123f). I've also tried, in addition, to apply only generic production modifiers (i.e. not Stone, Marble) when hurrying a wonder, but this can, again, be bypassed through overflow: when the overflow is added to the wonder, the extra modifiers are applied.
	In theory, overflow generated through Slavery could be subject to an additional penalty, e.g. reduced to 2/3, but I've found that too difficult to implement.

064d	Invalid orders don't create overflow
AdvCiv	*BtS*
When the production order of a city becomes invalid during the city owner's turn, the city	The validity of all production orders is verified at the end of turn – too late to ask for new orders

immediately asks for orders. If the turn is ended without giving new orders, the city's production yield is lost. (As far as I'm aware, the only way how a city's production can become invalid during its owner's turn is through a lost resource, a state religion change or a rebased air unit – see 001b –, so only these conditions are checked.)	from a human owner. Cities whose production order is invalid produce overflow production. Overflow produced in that way is not subject to any limits; can be stacked up arbitrarily. (No change:) When a player ends his/her turn without assigning a new order to a city that has finished its last order, the production yield is lost.

See also	The exploit for stacking up production is described here on CFC.
Rationale	Easy enough to prompt the city owner for new orders. The popup can't be overlooked, so, if the owner refuses to give new orders, it's fair enough to assume that the city is deliberately ignored and to discard the yield. Don't have to worry about excess overflow this way.
Tbd.	Not sure if I've implemented this correctly; I don't fully understand the code for updating plot groups. See comment in `CvPlotGroup::recalculatePlots`. Seems to work pretty reliably though – after having fixed a bunch of issues with my original implementation. (In particular the bug reported here; the attached savegame could be useful again.)

065	"Show Culture Turns" and "Commerce Subtotals" no longer optional
AdvCiv	*K-Mod*
The turns until next culture level (and thus border expansion) are always shown on the culture bar of the city screen. Commerce subtotals (e.g. "base research") are always shown in the commerce breakdowns of cities with specialists or other direct sources of special commerce.	There's an option for this, which is enabled by default in BUG, K-Mod and probably most other BUG-based mods too. Optional in BULL (called "Base Commerce" in the code) and enabled by default (like almost all options), included in K-Mod but disabled by default.
Rationale	To unclutter the BUG menu, especially the "Misc." categories.

066	Saved space on the Foreign Advisor screen
See also	073 changes the layout of the "Resources" tab
AdvCiv	*BtS*
No leader names are shown above the entries of the "Active" and "Info" tab. Moved the "Active" and "Info" tab up to the same height as the "Tech" tab, halved the horizontal margins and reduced the height of the bottom margin to 25%. The changes to the "Info" tab are only relevant if BUG's "Enhanced Info" tab is disabled (which is the case by default since AdvCiv 0.95).	In addition to a leader portrait on the left, the name of each leader is displayed above the portrait. The leader name also appears (no change in AdvCiv) when hovering over the portrait. The "Active" and "Info" have a peculiarly thick bottom margin.
Rationale	Especially vertical space is valuable (even on large displays).
The height of the main panel on the "Active,"	The panel takes up the entire height of the

"Info" and "Glance" tabs is reduced when there are few civs to list.	screen (apart from the margins), even if there is just one foreign civ to display information about.
Rationale	The large panels look inelegant.

067	BUG era display uncoupled from NJAGC (game clock) and other changes to NJAGC	
See also	[002k](): Year notation (AD/BC, CE/BCE) can be switched on the "Time" tab.	
AdvCiv		*BUG*
The upper half of the "Time" tab, which concerns the era display, does not require NJAGC to be enabled. The options on the lower half have been rearranged and a second dropdown menu has been added that allows the alternating views to be displayed for unequal amounts of time. Default settings revised so that the game turn is shown for a longer duration than the time of day (but the whole clock is still disabled by default). Removed the option to show the game turn progress as a percentage.		The tab is called "Clock". The switch for enabling NJAGC (Not Just Another Game Clock) is in the upper left corner and affects the era display and the clock settings – can't have the era without the clock. That said, the color settings apply to the era shadows on the tech tree even if NJAGC is disabled. By default, if NJAGC and the alternate display are enabled, the display switches every 5 seconds between game turn - slash - turn limit and game progress as a percentage; the time of day is always on display.
Config	I've added an option for displaying the game era instead of the active player's era. Then I realized that this would be a cheat as the game era can't be easily derived from the game state until all civs have been met, so I disabled the option again. Can be re-enabled through `BugNJAGCOptionsTab.py`.	
Rationale	Personally, I want neither an era display nor a clock, but there's no reason to couple the two. The era display is even in a different corner of the screen. The progress percentage seems entirely pointless to me; the other options I can see some use in: Showing e.g. "126/750" instead of "Turn 126" can serve as a reminder that the game is on Epic speed. I could also imagine that some players want to hide the game year, e.g. to make room for the time of day. The alternating display was perhaps not a great idea originally, but is kind of useful in AdvCiv because of the increased font size ([002b]()); the BtS clock doesn't fit on the panel anymore. With the second dropdown menu I've added, one can also build a sort of nagging clock that appears every 10 minutes just to remind the player that time is passing. That's a bit more subtle than the BtS alarm function.	
Credits	TheLopez is the author of the NJAGC component.	

068	BUG options can cause tech tree to be updated	
AdvCiv		*BUG*
When a BUG option that affects the visuals of the tech tree is changed, a dirty-bit is set and the whole Tech Advisor screen is reconstructed the next time that it is opened and then reflects the changed settings.		The "Tech Era" option that displays colored shadows on the tech tree and the options on the "Clock" tab that set the shadow colors usually only take effect after a restart because BtS (`CvTechChooser.py`) caches the tech tree widgets when tech tree is opened.

	The "GP research" works without restart.
Rationale	The BtS caching can't just be disabled. Opens the tech tree about one second faster (in a Debug build) and keeps track of the horizontal scroll position.

069	Changes to plot lists and the PLE (Plot List Enhancements) component
Credits	BUG lists 12mokeys and EmperorFool as the main authors of PLE.
See also	PLE has a function that predicts the vertical space needed for help text (`getTextLines` in `PLE.py`). This was sometimes underestimating the space, presumably because of AdvCiv's larger fonts, so my adjustments in the PLE code are tagged with 002b. That said, sometimes too much space was allocated (and I've only made that worse), so I'm not sure if this is entirely a matter of font size. Anyway, too much space isn't so bad, but too little space had meant that not all text was readable.
Tbd.	Often, no help text is shown for plot list icons on the city screen. This error occurs regardless of the plot list drawing method (vanilla/ PLE/ BUG). The handler for `WIDGET_PLOT_LIST` gets called and composes the text correctly, but the text isn't displayed. When loading a savegame and entering a city screen without selecting any unit first, it seems to work correctly; so perhaps some of the code in `CvMainInterface.interfaceScreen` (`screen.addCheckBoxGFCAt`?) needs to be repeated in the `updatePlotListButtons` functions (near `screen.changeImageButton`) when the city screen is up.

AdvCiv	*BUG*
Rearranged the menu items and rewrote much of the help text on the BUG menu tab to make the dependencies and side-effects between the PLE options clear. Also changed some of the in-game help text to match the terminology used on the menu.	Originally, all the options on the "Plot List" tab required PLE to be enabled and to be in charge of drawing the unit icons in the plot list. The BUG team changed this gradually so that some options can also work independently of PLE, but they didn't update the menu (or only in part).
Rationale	The PLE component is pretty complicated and mostly aimed at games with very large stacks, but some of the functions are actually very simple and universally useful, so it's good to have this sorted out.
PLE in-game help text shows turns-to-heal on a separate line and spelled out.	Shown without explanation on one line with the unit's strength and movement points.
Rationale	Too difficult to guess what that number means; I had to look it up in the manual. And vertical space isn't really (or shouldn't be) scarce.
When listing unit abilities, only the `BasicUnitHelp` string is displayed.	The full `UnitHelp` string is displayed (under a heading "Unit Specialties"), including a list of unique replacements and the production cost.
Rationale	Confusing clutter

Fixed a bug in `AStarTools.py` that had caused the PLE movement highlighter to crash sometimes.

Fixed a bug in `PlotListEnhancements.py` that had prevented bar colors from getting updated when changed through the BUG menu.

Fixed a possible bug in `PLE.py` that had caused health bars to be shown for civilian units. (Perhaps was intentional.)

Fixed a bug in `PLE.py`: The names of unit owners had been shown in random colors.

Fixed a Python crash in `MonkeyTools.getPlotHealFactor`. Credits: keldath ([CFC post](#))	

Adopted a few of the help text changes in PLE into BtS (so that they apply if PLE is disabled):

The help text for a unit icon doesn't show the unit owner if the unit is owned by the active player. The remaining movement points are shown (in addition to the maximum). XP is shown without parentheses (also in tile hover text).

Rationale	The game shows unit icons grayed out for units that belong to other players, and it's pretty rare anyway that units of different players reside in the same tile. Therefore the owner isn't important information in this context. Without the owner, the name, power, moves, XP and promotions often fit into a single line.
	Remaining movement points should be shown in order to match the information on the info pane.

If the BtS drawing method is enabled, no indicator circles are shown for foreign units unless they're wounded and the wounded indicator is enabled or have an attached Great Warlord and the Great Warlord indicator is enabled or when playing multiplayer with simultaneous turns.	Not sure about the PLE method and simultaneous turns. In singleplayer mode, BtS and BUG display an indicator circle regardless of the unit owner. For foreign units, it's always a white circle – indicating that the unit isn't currently expecting orders – unless the unit is wounded or has a Great Warlord (if the respective options are enabled).

Rationale	If the circle has no information value, it shouldn't be shown.
Tbd.	Disable the circle also when the BUG drawing method is enabled.

070	Gold rate color options
AdvCiv	*BUG*
Three dropdown menus on the "General" tab of the BUG menu for setting the text color of the gold rate when it's positive, negative or negative and exceeding the reserves in the treasury. The default colors are as in BtS. Update: Removed the menu for the "funds exceeded" color because the menu was getting crammed.	Checkbox "Gold Rate Warning" that shows the gold rate in yellow when it's negative but not exceeding the gold reserves in the treasury.

Rationale	I'm not sure what the idea behind the BUG option was. It adds code (which I haven't changed) for showing negative gold reserves in red, but I'm not aware of an exploit that allows negative gold reserves, and I see no need for a warning about that. That aside, one could argue that a negative gold rate shouldn't be shown in red because it's not an alarming condition (though it's certainly bad in a sense).
	I was going to set the color for positive gold to yellow to make gold rate and research rate easier to distinguish. Now that it's implemented, the yellow is a bit too unfamiliar for my taste (or just catches the eye too much). Also, if the gold rate is shown in the gold/ commerce color, the research rate should really be shown in the research color, and that's medium green (`COLOR_GREEN_TECH` I think); currently, the research rate is light green (`COLOR_POSITIVE_TEXT`).
See also	004 adds light green and light red (`COLOR_POSITIVE_TEXT`, `COLOR_WARNING_TEXT`) to the color palette of the BUG menu (through `Config\init.xml`).

071	Message on first contact

AdvCiv	*BtS*
When a rival is met, the game displays a message about this on the main interface along with an indicator at the tile where the meeting occurred (unless contact is made indirectly, e.g. through a vassal agreement).	No such message. If an AI rival is met, a diplo popup is shown. No notification about meeting a fellow human.

Config	Option on the "Alerts" tab of the BUG menu. Also allows the diplo popup to be disabled.
Rationale	It's often difficult to tell where a rival was encountered. The rival unit could e.g. have moved out of sight again or could be stacked with units of a third party. The option to disable the (rather pointless) diplo popup is a nice bonus. I'm not disabling them by default, in part, because I like the introduction texts from the Actual Quotes mod.

072	Changes to the "Current Deals" list (trade table)

AdvCiv	*BtS*
Hover text for trade items listed under "What are the current deals we have together?" shows the deal that the item belongs to (e.g. "Banana to Peter for Rice and 2 gold per turn" when hovering over "Banana") and whether/ when that deal can be canceled. Trade denial information is not shown in the hover text. This was tricky to implement because the trade table is not part of the SDK, and the DLL function that composes the help text (`CvDLLWidgetData::parseTradeItem`) is only given information about the trade item, not the CvDeal object.	It can be difficult – sometimes impossible – to tell what deal a trade item belongs to because given and received items are shown in separate columns and those belonging to the same deal don't necessarily align. The hover text says nothing about the deal either. Instead, for resources, the hover text says if the AI would agree to another trade for this item – usually, it won't because one resource of a type is enough.

Tbd.	Align the two columns by adding empty lines so that items of different deals are never shown on the same row. Adding newlines in `CvPlayer::getItemTradeString` can't accomplish this (the GUI won't show the newlines), but one could use a new type of nameless dummy trade item. Such items would have to be inserted in `CvDeal::addTrades` – I've tested this and kept the code in my (offline) archive. Taking care of all the side-effects is going to be tedious though. In particular, the code currently identifies gifts by checking whether one of the item lists is empty. Would have to look at all `getLengthFirstTrades`, `getLengthSecondTrades` and `CLinkList::getLength` calls. There might also be issues within the EXE (but I doubt it).

AdvCiv	*BUG*
After each trade item listed, if the respective deal can't currently be canceled, the number of turns that remain until cancellation is shown in parentheses. (Disabled by default.)	BUG has an option ("Deal Turns Left" on the "Advisors" tab) for this, but that only applies to the "Active" tab of the Foreign Advisor screen, not the trade table.

Config	I've extended the "Deal Turns Left" option so that it can either apply to the "Active" tab, the current deals on the trade table, both, or neither. The default is neither.
Rationale	In both contexts, hover text shows cancellation info; that should be enough. Gold-per-

	turn items already show the amount of gold in parentheses, and the double parentheses look a bit strange, e.g. "Gold per Turn (3) (6 Turns)". It's also not totally obvious what this means, i.e. that the first number is the amount of gold and the second the turns until cancellation.

073	Trade help on the "Resources" tab (Foreign Advisor)
See also	004w revises the help text for resources. 036 makes trades with non-surplus resources more common, and thus increases the importance of the "Resources" tab. And shows in help text how much gold the AI will pay or demand for a resource.

AdvCiv	*BULL*
Trade denial hovers merged from BULL. The changes in the DLL are tagged with "BULL - Trade Denial" (almost) as in the BULL code. Functional changes on my part: The hovers aren't optional, and no denial info is shown for resources if the reason is "you must be joking." Show "refuses to talk" as the denial reason when a rival refuses to talk and no other reason applies.	They use the tag "*BUG* - Trade Denial". Option on the BUG menu. At least in BAT, it looks like must-be-joking resources aren't shown on the "Resources" tab at all, not sure if that's standard BULL behavior or somehow customizable. (Or perhaps only K-Mod includes the must-be-joking resources?) The trade denial hover text ignore refusal to talk.

Rationale	There's really no downside to the extra help text, hence no need for an option.
See also	074 excludes must-be-joking resources from the trade table.

Enabled BULL's "Kill Deal" functionality. (I only had to remove a check for the presence of the BULL DLL for this, so it looks like this BULL feature could actually have been a BUG feature.) I've added the deal string (e.g. "Wine and 2 gold per turn to Peter for Sheep") to the help text.	Resource trades can be canceled from the "Resources" tab by clicking on a resource icon in one of the "Active" columns. Help text for those resources says "click to cancel" or how many turns remain until cancellation becomes possible.

Rationale	Without the deal string, it's not always clear which items are going to be canceled.

AdvCiv	*BUG*
Changed the layout of the "Resources" tab a bit. The column headings are now "Will Import," [gold icon] (available gold per turn), "Will Export," "Won't Export," "We Don't Need," "Importing," "Exporting," [gold icon] (trade balance). I.e. all from the other civ's perspective except for "we don't need." The columns are still grouped through dark background panels, but the group headings have been removed.	The group headings are written from our perspective: "Export," "Import," "Active," and the subheadings from the other civ's perspective: "Will Trade," "Won't Trade," (again) "Will Trade," "Won't Trade," [gold icon], "Import," "Export," [gold icon].

Rationale	The two different perspectives are confusing, and the removal of the group headings frees up space for a larger "Surplus Resources" box (see below).

The column for resources that the other civ is unwilling to import has been removed and the column for resources that the other civ is unwilling to export has been split into "We Don't Need" (resources that we already have – denial reason: "you must be joking") and "Won't Export"	The "(we can) Export, (but they) Won't Trade" column contains the resources that both civs are able to trade. The "(we can) Import, (but they) Won't Trade" essentially also contains those resources plus some resources that the AI is unwilling to trade for other reasons.

(all other reasons).	
The "Surplus Resources" box has been renamed to "Domestic Resources" and lists all resources that we're able to trade. Surplus resources are listed first and can easily be recognized by the yellow number that says how many copies we own; that number isn't shown for non-surplus resources.	Only surplus resources are listed in that box. All our non-surplus resources appear multiple times in the "Won't Trade" columns though.

Rationale	The description on the right should make clear that the BUG layout is redundant. The information in the "We Don't Need" column still overlaps with the "Domestic Resources" box, but, without that column, a complete list of foreign resources would be missing.
Tbd.	When the player has surplus resources of more than 28 resource types, the first row of the "Domestic Resources" box is going to be too small, and icons are going to be shown outside the box or even off-screen. This is because I don't know how to display multiple rows of resource counts (the yellow numbers). Not an urgent problem because 28 is a lot when there are only some 35 resources in the game, and on widescreen displays, all resources might fit in a single row. On that note, the code should check whether all resources (incl. non-surplus) will fit into a single row – easy to do – and show the second row only when needed – not quite so easy.

074	Some must-be-joking items excluded from the Trade screen
Rationale	If it's not obvious why these items can't be traded, then the explanation "Surely, you must be joking" isn't going to make it more obvious, and beyond the explanation text, untradeable items serve no purpose on the Trade screen. That said, in some cases the omission of an item could appear like an error, so I'm not removing all of them.

AdvCiv	*BtS/ K-Mod*
The Trade screen doesn't show must-be-joking • resources, i.e. resources that the other side already has (and doesn't need for a corp.) and • civics, i.e. civics from the first row. Exception: When a resource deal is canceled on account of "must-be-joking", i.e. when the recipient of a resource acquires direct access to it, then the canceled resource is not hidden in the diplo popup for renegotiation.	K-Mod excludes "cold, dead hands" cities and treats unrevealed cities as secret. When an AI civ cancels a resource deal with a human civ, a diplo popup for renegotiation is shown at the start of the human turn. That popup restates the terms of the canceled deal.

See also	See the last quote box of this post about the exception. The known issue is too much work to fix because it involves the EXE. If all AI diplomacy is moved to the start of the human turn (see 001e), then the problem will go away.
	If an AI civ cancels multiple deals at once, 133 shows all canceled trade items in a single popup. 074 doesn't support multiple popups in a row.
	073 makes sure that the resources of all known players are visible on the "Resources" tab of the Foreign Advisor. (One could argue that unrevealed resources that the AI isn't offering for trade should be secret, but this would take some effort to implement.)

075	Units in cargo wake up when the ship has reached land and is out of moves

AdvCiv	BtS
When a ship has spent all moves, has no further moves or other missions queued and is in a land tile (city, Fort) or adjacent to land, any land units in cargo that are able to move are woken up. Units that have been given any order while in cargo will not wake up. (Technically, units are set to a new activity type called "boarded" when they go aboard, and they have to be in that state in order to wake up.) All of the above only applies when automatic unit cycling is enabled (it is by default).	Units are set to the "sleep" activity when they enter a ship. I don't think they ever wake up from that automatically.
Rationale	Because of automatic unit cycling, it's easy to forget that units can disembark when a ship reaches its destination with its final movement point. The cargo units then lose an entire turn. I'm limiting the wake up conditions as much as possible because units asking for orders prematurely can be annoying. Hence also the restriction that an embarked unit will only ask for orders once.
Tbd.	Perhaps an option "Wake Up Cargo When Reaching Land" on the "General" tab under "Commands." Hopefully not needed. GlobalDefines option? It might be that units are distributed rather unintelligently at times when a stack of land units is moved onto a stack of transports. [C2C thread](#)

076	Player options menu revised

AdvCiv	BtS
Renamed the first tab from "Game" to "Controls". Most of the changes to text were only made in English and German.	By "player options menu", I mean the one that gets opened by Ctrl+O. BtS doesn't seem to have a distinct name for that menu. The settings there only affect the active player.
	On modding this options screen: Unlike the Custom Game screen, the files are mostly in the SDK, so it's easy to modify. The only obstacle I encountered is the inflexible `resetOptions` function of `CyUserProfile`, which isn't in the SDK.
	Upon exiting the game, most of the options are stored in a 200-byte `.pfl` file in `AppData\Local\My Games\Beyond the Sword\Profiles` — either in `Default Profile.pfl` or in a custom file created by the user through the "Other" tab of the player options menu. (Unless that pfl file is set to read-only through Windows file properties. That's a way to make the options revert to a fixed configuration at program start.)
Rearranged, renamed and, in two cases, hid items on the player options menu, however, without changing the format of the options profile	The pfl file is written in a binary format that will break when options are removed from or added to `CIV4PlayerOptionInfos.xml` or

or the order of options in XML, so that these changes won't cause any compatibility issues. Moved some of the checkboxes from the right column of the "Graphics" tab into the left column, mixing them with the dropdown menus. E.g. the "No Movies" option is now placed above the "Movie Quality" dropdown and "No Buildings" (in Globe View) under the "Globe Quality" dropdown. Removed the "Graphics Level" dropdown, meaning that the setting will remain at whatever value is stored in the player's profile.	`CIV4GraphicOptionInfos.xml`, unless blank options reserved for mods (`PLAYEROPTION_ MODDER`) are used. "Graphics Level" doesn't seem to have any effect of its own. Looks like a change in that dropdown merely causes several other settings to change, namely the other three "quality" settings, anti-aliasing, "Low-Res. Textures", "No Movies", "No Buildings in Globe View", "No Effects", and "Animations Frozen". They probably forgot "High Detail Terrain", which was added by patch 1.61.
Rationale	A widget that will overwrite several other settings is similar to a reset and that would have to be very clearly communicated. However, on current hardware (say, post-WinXP) most of the settings tied to "Graphics Level" hardly affect performance (or even power consumption), so it's better (and easier) to remove this master switch entirely.
Config	Can always change the "Graphics Level" by unloading AdvCiv.
Tbd.	Find out if "Render Quality" has any effect. I'm seeing no immediate effect, and, after restarting, the terrain always looks a little different, so it's difficult to be certain that there is no change. There could well be some very minor impact, or a noticeable impact only in combination with other settings, e.g. only at a low resolution. On the web, I've only found a few baseless claims. If there is no appreciable effect, then it should perhaps be force-set to "low" (to make sure that no computing time/ power is wasted) and the dropdown removed. Currently, my tooltip says "Effect Unknown". And test some very high resolutions to see how the fonts and icons scale; then possibly add a warning about that to the "Screen Resolution" tooltip. I'll need a larger monitor for that.
The space cleared in the right column of the "Graphics" tab allowed me to add headings to structure the remaining checkboxes, to move two options from the "Controls" tab to the "Graphics" tab that might as well be graphic options ("Quick Moves" and "Numpad Help") and then to add headings to the "Controls" tab as well. Rearranged the checkboxes on the first two tabs and made some (mostly minor) name changes, in part to take advantage of the headings; e.g. having a heading "Automation" means that "automate(d)" doesn't have to appear in all of the option names.	The checkboxes on the first two tabs are haphazardly arranged and not structured by any headings.
Rationale	Regarding "Quick Moves", the options for unit controls on the "Controls" tab and those for animations on the "Graphics" tab are both concerned with unit animations. Putting them all on one tab might be nice, but "Quick Defense" and "Show Friendly Moves" need to be on the first tab for easy access, and there isn't enough room there for all of them. Even moving only "Quick Moves" and "Combat Zoom" to the "Controls" tab would require a major reshuffle. And these two really are just visual changes, whereas most options on the "Controls" tab allow a tradeoff between optimal play and pace, e.g. the automation options, "Wait at End of Turn", "Stack Attack" and I'd say also "Quick Defense" as this option makes it sometimes impossible to keep track of all enemy attacks.

Changed almost all tooltips to provide more precise information and, in some cases, hints and recommendations. Added tooltips to the dropdown menus and to the reset and exit buttons.	Many of the tooltips more or less just repeat the name of the option, which is usually not self-explanatory (certainly not for new players). No tooltips for the dropdowns and buttons.
Added a "need restart" popup to the "Low-Res. Textures" option and the "Globe Quality" menu, and added info about having to restart or reload to the tooltips of any options that need it.	Restart popup for "Fullscreen", "Single Unit Graphics", "High Detail Terrain", "Graphics Level", "Audio Speakers".
Removed the banner from the "Audio" tab; the cleared space remains mostly unused.	A banner of a sound card vendor (sponsor?) is shown on the "Audio" tab.
Device names in drop-down menus shortened as needed to make sure that the menu fits in the window. Moved speaker settings and custom music folder to the left column.	I never even knew that a custom music folder can be configured because the button had been off-screen.
Removed the panel for network settings from the "Other" tab. That means, the connection speed set in the profile is assumed.	The only network setting is "Modem" vs. "DSL/Broadband". Not sure if this has any impact on networked multiplayer games.
Rationale	No one has a non-DSL modem. Can still change this through "Bandwidth" in the INI file. With this panel removed, the Profile panel is the rightmost one, which makes sense: after configuring everything, the player may want to create/save a profile. Also, in BtS, one can get the impression that a profile is just some sort of user id for network games. Regarding the layout of the "Audio" tab: It's better to have voice chat on the lower right than speaker settings and custom music folder because the voice chat options are a bit complicated and rarely used.
Tbd.	Should perhaps mention in the tooltip for "Show … Moves" that those options have no effect when playing with simultaneous turns.
Changed the default values of several options (at my discretion).	
Rationale	Doesn't matter much because the BtS defaults are set in the profile upon installing BtS and, when a mod is loaded, the values in the profile are used. The defaults do matter when a player manually resets the profile, or if it is automatically reset, which happens under some rare error conditions. Also, the defaults can be looked up in XML as a reference.
The reset button on the "Graphics" tab doesn't affect the "Fullscreen" checkbox, nor the "Screen Resolution" and "Menu Background" dropdowns. The quality dropdowns are set to "high" and MSAA to 2 samples. The reset button tooltips explain what gets reset and how that will affect the current profile. The need-to-restart popup is only shown if one of the options that was changed requires it. Removed the reset button on the "Other" tab.	Each of the four tabs has its own reset button that affects all settings on that tab except (I think) "Screen Resolution". The default values for the checkboxes on the first two tabs come from the ...`OptionInfos` XML files, about the others I'm not sure. For the "Graphics" dropdowns, the same settings as for medium "Graphics Level" seem to be used, i.e. 2 MSAA samples and all levels at medium. The effect on the menu background dropdown seems erratic. When "Graphics" or "Audio" are reset, the need-to-restart popup is always shown.
Rationale	It should be obvious that the game is normally played in fullscreen. Perhaps a sensible resolution could be set based on the resolutions supported by the hardware, but that might take some work to implement. The menu background is a matter of taste.

	With only the Clock options left, a reset button on "Other" is unnecessary, and could be misunderstood as affecting all tabs.
Renamed the "Detailed City Info" option to "Numbers on City Bar", and it only shows production turns and turns to grow.	Also shows a research icon on the billboard of the city with the highest research rate, a gold icon for the highest gold rate and a production icon for the highest production rate.
Rationale	The meaning of the icons is obscure and, as the capital usually has the top gold and research rate, they're quite unhelpful except possibly for the production icon. Without the icons, the option makes sense on the "Graphics" tab because the turn numbers are really just a visual aid for players who find the one-turn projections always shown on production and food bars insufficient.
See also	002f allows the icons to be enabled through the BUG menu.
Tbd.	Would be nice to add a tab for controlling some important settings in `CivilizationIV.ini`. To get those settings more visibility and to allow them to be changed from within the game. (For some, it may be impossible to apply them immediately, but the options screen has warning tooltips and popups for such cases.) Wish list: `MouseScrolling`, `MinimapTrilinearFilter` (some might prefer it if they try it), `SetMaxFrameRate` (unlimited frame rate might be a bit wasteful for a turn-based game), `HidePythonExceptions`, `MaxAutoSaves`, `AutoSaveInterval`, `CheatCode`, `ScreenHeight`, `ScreenWidth` (for custom dimensions in windowed mode; don't know if they could be immediately adopted), `LoggingEnabled`, `RandLog`, `MessageLog` (the log settings would have to be given clearer names). Example of how to change an `.ini` setting from within the DLL: "We the People" Git commit by Nightinggale The "INI" tab should also show the path of the `.ini` file to which the settings will be written.

077	Changes to "Demographics" tab (Info screen)
See also	Info graphs: 004s, 091
Credits	This CFC post (by Long try) demonstrates that the BtS screen leaks information in the early game.
Tbd.	Refactoring – there is a lot of redundant code in `CvInfoScreen.py`, pretty much everything times 9. Needs a "Demographic" class that "crop yield", "soldiers" etc. can be instances of.

AdvCiv	BtS
The columns "best rival" and "worst rival" show the value, name and rank of the best or worst rival whose demographics are visible through espionage. Alternatively, as an option, a "?" can be shown in the "best" and "worst" column when the demographics of the best or worst rival aren't visible. Or, as another option, the name of the best or worst rival can be shown even if its demographics aren't visible. A "?" is then displayed if the best or worst rival hasn't been met.	Those columns show the value of the best and worst rival regardless of espionage or whether that rival has even been met. Only the value is displayed though (e.g. the number of soldiers). The main point of these columns was, I suspect, to provide points of reference for the player's demographics.

Config	The "options" are just boolean flags near the top of `Assets\Python\Screens\CvInfoScreen.py`.
Rationale	See under *Credits* above. I particularly don't like that the player can tell on turn 0 that some unmet rival has the Charismatic trait. Giving away any concrete rival values is problematic unless those values are revealed through the graphs anyway, i.e. when demographics are visible through espionage. Potential problem: If a player focuses all his/her espionage on one or two targets in order to reveal their research goals, then the rival columns may not contain much useful information. I've tried a less restrictive approach (still in the code as an option) that also doesn't give away values but names the best and worst team regardless of espionage; however, I worry that this could incentivize players to check the "Demographics" tab every few turns.

The rival average is rounded to the nearest multiple of 5, no longer includes vassals and teammates of the active player and isn't shown when there are fewer than 4 rivals alive. Minor civs are also excluded.	The average is computed over all civs except the active player. It's rounded to the nearest smaller integer and shown regardless of the number of civs alive.

Rationale	The (more or less exact) average of land tiles gives away how many civs start at the coast, and the other values probably aren't safe either. Moreover, the changes from turn to turn could reveal information throughout the game. Coarser rounding appears to solve these problems, and the resulting approximate average is still useful (at least by the midgame) for putting the player's values into perspective.
	With just 3 rivals, if best and worst are known, the one in the middle could be computed from the average, which is tedious. Hence no average when the sample size is small.
Credits	Exclusion of minor civs adopted from "Dawn of Civilization" ([Git commit](#))

Removed the final row (Export-Import). It can be re-enabled through an option (see *Config* above), but this will result in a vertical scrollbar since there is only enough space for 8 demographic values now. If it's enabled, the value is the player's total commerce from foreign trade.	The value is the player's total commerce from foreign trade minus the sum of the commerce that other civs receive from the player.

Rationale	Seems like the least useful statistic. It doesn't make much sense to subtract other civs' commerce because Civ 4 isn't a zero-sum game (unless only two civs remain); it's mostly flavorful (trade balance is a widely known economical concept). Commerce from foreign trade is still shown on the Finance tab (Economics Advisor).
See also	[CFC post](#) by a player confused about the significance of the Export-Import stat.

Layout changes: Except for rival average and the first (title) column, all columns have double rows with one item on top and another at the bottom.	Only the first column has double rows, showing the name of the value (e.g. "crop yield") on top and the unit of measurement at the bottom (e.g. "million bushels"). All the other columns just show a single number.
The worst/ best rival columns show the value on top and the rival name and rank on the bottom. The (player) value and rank columns have been merged into a single column with the value on top and rank at the bottom.	
Changed most of the labels to use the available space efficiently and resized the columns. Also using some icons. Shortened a couple of leader name translations (French, German, Spanish) to make them fit into the columns.	No icons; all text.

Rationale	Putting the rival value, name and rank all in one table cell would be difficult to read. Since the average column has only a (numeric) value to show and it would look strange to leave the top cell empty, all values have to be shown on top so that they're easy to compare (and all ranks at the bottom). Since the unit of measurement should be on the same row as the value, I've put the units on the same line as the names of the values. That's a bit tight, but it fits.
Decimal ("thousands") separators added.	
Rationale	To make the large numbers easier to read.
When the Info screen is shown after winning, losing or retiring, all info on all tabs is revealed as if in Debug mode.	Same except that the BtS Demographics tab didn't have anything to hide. And all info continues to be revealed after selecting "just one more turn" (extended game).
Rationale	Don't want additional info to be revealed in the extended game; the player can enter Debug mode for that.
The yield statistics about rivals are moving averages.	
Rationale/ See also	To be consistent with 004s, which shows moving averages on the "Graphs" tab.

078	BUG/BULL GP bar options
See also	001c fixes the birth probability display. 064 adds a separate option for the hurry tick marks.
Tbd.	Add the GP icons to `Res\GameFont.tga` (`GameFont_75.tga`?) and let the init function of `GPUtils.py` use those instead of the yield icons.
AdvCiv	BUG
Option added to show the GP bar and GG bar only once there has been any progress. More specifically, if the option is enabled, the GG bar appears once any XP has been gained, and the GP bar appears once any (non-GG) GP points have been generated by a city. All the bars and the new option are disabled by default.	If the GP or GG bar is enabled, the bar appears right at the start of a game although, typically, no XP is gained for the first 15 turns, and no GP points for the first 50 turns.
Rationale	Not enabled by default because the sudden appearance of the bars could be confusing when a player hasn't explicitly enabled that behavior.
The Great General bar gets updated when combat XP changes, i.e. this sentence in the BUG help file no longer applies: "*One thing to note is the display of the bar will not automatically update after a battle, but you must enter a screen and exit out for it to display properly.*"	
If tick marks are enabled, overflow research is shown as part of the semi-transparent portion of the research bar.	Overflow research is always shown as part of the solid portion of the research bar and thus indistinguishable from the current research progress. The semi-transparent portion shows the per-turn research rate without overflow. However, the production bar on the city screen shows overflow (only) in the semi-transparent portion.
Rationale	Solid overflow suggests that the overflow has already been spent, but it isn't spent until end of turn (because it's really part of the research rate). And it's inconsistent with the

	production bar. I'm keeping the BtS behavior if tick marks are disabled because the per-turn research rate without overflow is an important statistic that should arguably be visualized somehow, either through the semi-transparent portion or, preferably, through tick marks.

AdvCiv	*K-Mod*
"Great People Rate Breakdown" merged from BULL, but without an option, i.e. the hover text is always enabled. It shows, when hovering over the GP bar on the city screen, how many GPP are generated by buildings and how many by specialists.	Unavailable (not merged from BULL)

079	AI greetings

AdvCiv	*BtS*
Only about half of the AI leaders brag about their best unit (based on `CONTACT_DEMAND_TRIBUTE`). When a braggart AI has a new best unit, it brags about that unit at the first opportunity (unless the best unit is known anyway because AI technologies and resources are visible to the human player) and, if contacted again, with a random chance as in BtS.	If the AI doesn't warn about its worst enemy (see below), it brags about it's current best unit with a probability of 25% unless Pleased or less powerful than the player. The roll is repeated every time that the AI is contacted. AI personality doesn't matter.
Rationale	To eliminate the need for contacting the AI multiple times to find out its best unit in the early game. At first, I implemented a random chance of never bragging about a particular unit based on the `CONTACT_DEMAND_TRIBUTE` divisor, but, ultimately, I think it's better to make the early warmongers always reveal their best unit. (For some of the peaceful leaders, this would be out of character and shouldn't be needed either.) Of course, contacting the AI every few turns is still tedious, but I don't think the information is usually relevant, so I'm not implementing an alert message or sth. like that.
See also	005a changes the `CONTACT_DEMAND_TRIBUTE` divisor for several AI leaders. Most of the leaders with a sufficiently low divisor (less than 400) are listed there.

The AI remembers when it has warned a human player against trading with a particular enemy, and is more likely to warn if it hasn't warned before and less likely otherwise.	If there is a worst enemy, there's a 25% probability that the AI will warn about it when contacted. This is accompanied by an angry facial expression in BtS, and, in K-Mod/AdvCiv, by the trade-refusal expression.
Rationale	Mainly to reduce the frequency of the somewhat jarring stop-trading leader animations.
After the first era, the AI says "we would encourage you to stop trading ..." in the greeting only if the human player has annual deals with the worst enemy.	Will say "watch out for our worst enemy" only if the accumulated trade value in AI memory is 0, and otherwise, i.e. basically if the human has ever traded with the enemy, "we would encourage you to stop trading ..."
Rationale	To address a to-do comment by the original developers: "*maybe make this a little looser (by time...)*"

080	Warn about XP loss on upgrade	
AdvCiv		*BtS*
The upgrade button says how much XP will be lost (if any).		Units lose all but 10 XP when upgrading. Reportedly, the BtS manual mentions this but the in-game help text doesn't. And as no promotions are lost, it's easy not to notice.
Tbd.	Limiting the number of promotions that units can acquire is good, but I'd prefer a mechanism that somehow allows old promotions (which may not make that much sense on the upgraded unit) with new ones. Not sure how exactly.	

081	Naval AI for land wars (not much done yet)	
See also	905 increases the speed and bombardment rates of some ships.	
AdvCiv		*BBAI/K-Mod*
The AI may train a few ships with a transport or bombardment ability for land wars if the city that the AI wants to focus its attacks on is coastal. The AI may train more ships when there are hostile ships in its borders.		After I fixed a likely bug in BBAI code (see 017), the AI trains only a minimal number of warships except when planning a naval landing.
Tbd.	If naval bombardment is further buffed, an AI type `UNITAI_ATTACK_CITY_SEA` should be added. Should sometimes train ships for naval bombardment even when there is no war plan yet. To be added to BBAI's "city hunting stack" code.	

082	AI changes for naval assaults (few so far)	
Tbd.	AI shouldn't keep units for naval assault embarked; cf. this CFC thread. To get the AI to drop invasion forces in a friendly bridgehead (and to overcome the inability to attack landlocked enemies), it might suffice to do the following: In `CvUnitAI::AI_goToTargetCity`, check if the current unit is a sea unit with cargo, and if the target city is (about to become) hostile and in a different area than the unit. If so, find the reachable (in particular: coastal) non-hostile city in the target area that is nearest to the target city. If that city is not too far away from the target and nearer to the unit, then, with a high-ish probability, go to the non-hostile city. However, if the owner of the target city is expected to be short work, go directly to the target city if possible. Implement `UWAICache::updateRelativeNavyPower` (see comments there) and an auxiliary function for the intel ratio. Once there is a cached intel ratio, CvPlayerAI::AI_targetCityVal could also evaluate resources from the city owner's point of view so as to be able to "take the Oil." related CFC post `CvUnitAI::AI_assaultSeaTransport` should estimate a) (easy) the strength of the own escort (just add up the strength values) and b) the strength of defending ships that will intercept the group. Cheat by using the target's (unless barb/minor) UWAI fleet power – once i implement that `relativeNavyPower` function, the cheat will more or less go away. Also need to account for factors that may distract the defending fleet. Perhaps based on the number of the target's coastal cities plus half the number of own coastal cities (enemy could	

	use its fleet to harass those) plus one quarter of the number of coastal cities of our war allies.
See also	If [162](#) (crossing a border on the same turn as declaring war spends all movement points) is replaced with some form of ZoC rule, more AI changes will be needed.
	[040](#) deals with naval AI for founding cities and [046](#) with stranded units.

AdvCiv	*K-Mod/ BBAI*
When a group of AI units is supposed to start a war, the pathfinder avoids entering the enemy's border earlier than necessary.	Only avoids ending a turn inside the enemy's border. Doesn't avoid moving through.
Rationale	An oversight by the earlier modders. It would be even better to enter the enemy's territory without declaring war when it's clear that the units are going to leave and re-enter, but that would be a bit more work to implement.
In a non-"total" war, when a naval assault stack is large enough to serve as reinforcements but too small for an initial attack and no reinforcements are needed abroad, then the AI looks for a landmass with at most two hostile cities and launches an attack there if the assault stack appears to be powerful enough.	`AI_assaultSeaReinforce` was added by BBAI. Naval assault stacks consider that routine if they have at least as many units loaded as are required for a land-based AI city attack stack (`AI_stackOfDoomExtra`). An initial attack (`AI_assaultSeaTransport`) is only considered when the naval assault stack has at least twice as many units loaded as are needed for a land-based attack. (That's the intended BBAI behavior anyway; because of a bug – fixed by K-Mod – it hadn't actually worked that way.)
Rationale	`2*AI_stackOfDoomExtra()` is already ca. 10 in the Classical era. That's often overkill for attacks on enemy colonies.
See also	Discussion on CFC: [link](#)

083	AI changes for city attacks
See also	[001t](#) and [082](#) deal with the path that AI units take when declaring war.

AdvCiv	*K-Mod/ BtS*
Made sure that the AI doesn't blindly attack a city while trying to get to the best tile to attack it from.	There was an assertion saying "*no suicide missions*" that triggered sometimes.
Barbarians ignore units positioned adjacent to a city when deciding from which tile to attack the city.	[Reportedly](#), the BtS AI is generally unable to plot around enemy units; in K-Mod, I can only recreate this for Barbarians vs. units blocking the entire inner ring of a city.
Tbd.	Non-Barbarian AI stacks still disregard tiles with enemy defenders when deciding from which tile to attack a city. K-Mod comment: "*consider fighting for the best plot*"
	The choice of the tile to attack from needs to take into account how dangerous the enemy units are. Currently, once the AI has decided on an attack path, it won't enter the city directly even if all defenders abandon the city.

084	Solo games

Rationale	It's possible to close all except one player slot on the Custom Game screen – and that makes some sense too when a player just wants the map to him- or herself. So that that mode of play should be supported if it isn't too much trouble.

AdvCiv	*BtS*
No Conquest or Diplo victory in solo games.	If Conquest victory isn't explicitly disabled, the game ends shortly after the player founds his/her first city. Diplo victory impossible, but not explicitly unavailable.

Rationale	The other victory conditions make *some* sense without any opponents.
Difficulty factor for Hall-of-Fame score gets divided by 4 when playing without competition.	
Rationale	Don't want solo wins (especially not Domination) to end up at the top of the Hall of Fame screen.

085	Changes to the tabular scoreboard (i.e. only applies when "Tabular Layout" is enabled on the "Score" tab of the BUG menu)
See also	120h changes the meaning of the espionage column. For a scrollable scoreboard, this Dawn of Civilization pull request could be helpful. Though I don't mind the scoreboard taking up much of the screen height.

AdvCiv	*BUG*
When an underscore is placed before a letter in the "Column Order" formatting string, the respective column appears only if the "Expand on Hover" option is enabled and the mouse hovers over the scoreboard. By default, the trade network, Open Borders, Defensive Pact, power ratio and Golden Age columns are set to appear only on mouse hover, and "Expand on Hover" is enabled by default.	The formatting string on the "Score" tab of the BUG menu allows players to select which columns should be displayed and how they should be arranged. Underscores in the string don't have any meaning and are ignored.

Config	See the help text for the "Expand on Hover" option on the "Score" tab of the BUG menu. The letters with a preceding underscore should be placed on the left side of the scoreboard, at least to the left of the player score (letter "S") and civ display name (letter "C") to make sure that the score and name don't move away from under the cursor when the scoreboard expands. Previously (v0.95 and earlier), all the info in the expansion columns was disabled entirely by default, so it's all nonessential and players who find the expanding scoreboard distracting only need to toggle off "Expand on Hover". However, players who want e.g. the power ratio to be shown permanently, will have to edit the "Column Order" string to remove the underscore and may also want to move the power ratio column farther to the right. For reference, the K-Mod default display string was `!?2WSZVC6EPTUNBDRAHQ*LO`, BUG's default is `!WSZVC?EPTUNBDRAHMQ*LO`. The AdvCiv default can be found in `Assets\Config\Advanced Scoreboard.xml` under "DisplayOrder". (In case that it gets overwritten when importing settings from another BUG installation.) When no expansion columns are set in the display string, then the "Expand on Hover" option becomes automatically disabled. This is done to ensure that no time is wasted by the DLL on checking whether the cursor gets moved off the scoreboard. Especially relevant when a player imports (copies) settings from another BUG installation.

See also	[106d](#) changes the default formatting string for the scoreboard. Based on [003r](#) (code for deferred UI updates)
Rationale	To make some info about other civs quickly available without obstructing the view on the main map all the time. Choice of the extra columns: The trade network icon isn't helpful in itself, but the hover text (merged from BULL) lists trade routes and active deals.
Tbd.	In large games, unit animations (if enabled) generally stutter a bit while hovering over the expanded scoreboard. Camera movement (e.g. arrow keys) while hovering also isn't entirely smooth. I think that's just the result of recomputing and redrawing the scoreboard once per game update (i.e. every 250 ms). The only (potential) remedy I can see would be to write a (Python) update function that only reinitializes the scoreboard widgets. Cf. comments in `CvPlayer::setScoreboardExpanded`. The problem seems to get a bit worse when all columns are enabled. It gets much worse with more than 30 civs. [Bug report](#) It seems that sometimes (rarely?) moving the cursor onto the scoreboard right after it has collapsed causes the frame rate to drop 5 FPS, suggesting that the scoreboard is getting redrawn without intermittent game updates. Or perhaps it's just somehow a special case of the problem described above. Probably won't fix: When moving the cursor quickly from the expanded scoreboard to the flag button, the mouse focus switches to the main map behind the flag button. That's probably a side-effect of the `CvInterface::makeSelectionListDirty` call in `CvGame::update`. Another side-effect: When the mouse is moved across the scoreboard onto a Globe layer button and the scoreboard collapses while the mouse button is down, the click on the Globe layer button isn't registered. Probably can't be helped because the DLL can't tell whether a mouse button is being held down.

During diplomacy the scoreboard can be expanded, but will then remain expanded until diplomacy ends.

Rationale	Workaround for a problem with flickering hover text; see comment at the start of `CvPlayer::setScoreboardExpanded`.

AdvCiv	*BtS/BUG*
Column added that shows the Golden Age icon when another civ is in a Golden Age or the negative-gold icon when in Anarchy. Help text for the icons says how many turns remain.	Golden Ages and revolutions are announced, but I'm not aware of any way to check – reliably – if an opponent is currently in a Golden Age or in Anarchy. Golden Age and Anarchy length are public knowledge though.

See also	[152](#) uses the "fist" icon for AI war trades. That icon has become strongly associated with AI war planning through the BUG mod. So that icon can't be used for Anarchy. And I like the negative-gold icon better anyway.
Rationale	For the active player, Golden Age and Anarchy are already prominently displayed in BtS; therefore show the scoreboard icon only for other players. The more often no icon needs to be shown (for any player), the better because then the entire column disappears.

Hover text for the power ratio column added. Explains what the ratio means and why rival demographics are visible: by how many espionage points the threshold for seeing demographics or research (whichever applies) is	BUG has added the power ratio column. No hover text. The power ratio is only shown if a rival's demographics are visible through espionage. In BUG, the power ratio column is enabled by default, in K-Mod, disabled by default.

exceeded. Power ratio column enabled by default but only appears when the scoreboard is expanded. For teammates of the active player, high and low power ratio aren't color-coded.	Color-coding is customizable through the BUG menu ("Score" tab) but the settings apply to rivals and allies alike.
See also	Color settings for power ratios (`Advanced Scoreboard.xml`) are tagged with [advc.004]().
Rationale	The power ratio is useful (especially because it implies visible demographics), but bulky and potentially confusing, in part, because it isn't obvious which power rating is in the divisor. Hover text and expanding scoreboard address these problems.

Tbd.	The info about the see-research threshold might make more sense in the rival-research help text. Would have to add a new help widget for that because the widget currently used doesn't have room for the rival player id in its data structure; but that's not much work. Could then also remove the active player's research progress from the help text (unexpected in this context). Still, showing the see-research info in the power ratio help text is quite economical as it doesn't take up any extra space (see-research implies see-demographics).
	
	While I wouldn't want to use the attitude and worst enemy icons even in the expanded scoreboard, it would still be nice to add some simple hover text to them:
	  Worst enemy: "You are Alexander's worst enemy"
	  Attitude: just "Friendly" etc.
	
	Some useful info that could be shown in hover text for the religion icons? No text at all currently and they're always shown.

Clicking the active player's name or score on the (maximized) scoreboard has no effect. (The scoreboard can still be hidden entirely through "Toggle Scoreboard".)	The scoreboard is minimized when the active player is clicked. The minimized scoreboard shows only the active player's row.
Rationale	I don't think minimized scoreboard is good for anything, and players who accidentally minimize the scoreboard may not know how to maximize it again.
See also	[Post]() on CFC asking how to maximize the scoreboard.

A score breakdown is shown only when hovering over a civ's score value. Always for the active player and, for others, in Debug mode if the Ctrl key is held down. When hovering over the active player's name on the scoreboard, help text shows whose worst enemy or war enemy the active player is, and the active player's total war weariness is shown. (I.e. pretty much the same info as is shown by K-Mod when hovering over the active player's portrait on the Foreign Advisor screen.)	When hovering over the active player's score or name, a score breakdown is shown. Can't show score breakdowns for other civs.
Rationale	Want to make worst-enemy info and war weariness easier to access.
Tbd.	Combine the functions `CvDLLWidgetData::parseContactCivHelp` and `CvGameTextMgr::parseLeaderHeadHelp`; they do largely the same thing.

AdvCiv	BtS
When rival research is visible, the scoreboard shows the tech progress as a percentage.	Show turns remaining.

Rationale	The turns remaining fluctuate too much as the AI adjusts its sliders or goes into Anarchy. Plus, the percentage can be shown even when no tech has been chosen yet, avoiding confusion as to whether research has stopped being visible. (Naturally, this problem could also be solved e.g. by showing some sort of empty tech icon when no tech is being researched.)

086	Hover text on the "Finances" tab (Economics Advisor) revised	
See also	Hover text merged from BULL is tagged with "BULL - Finance Advisor".	
AdvCiv		*BtS/BULL*
Mostly removed headings that didn't contain any additional information.		Some items in the commerce, income and expenses breakdowns have hover text with explanations or additional breakdowns.
Finance tab lists commerce from domestic trade even when it's 0.		
Rationale	To avoid confusion.	
Credits	Th334	

087	Streamlined some BULL hover text	
See also	086 above also deals with that. 004: Misc. UI changes 124 prevents trade with unrevealed cities. I've added a bit of help text about that to the BULL "Trade Hover". Only shown when 0 cities of a rival are known.	
AdvCiv		*BULL*
The list of active deals in the hover text of the trade network icon on the scoreboard omits all dual deals, i.e. peace treaties, Open Borders agreements and Defensive Pacts (DP).		BtS show no hover text for that icon. BULL shows information about foreign trade routes and lists all active deals.
See also	106d hides the trade network icon by default.	
Rationale	Open Borders can be inferred from the trade route info. For peace treaties, there's a separate icon (enabled by default). The icon for DP is hidden by default, but DP really have nothing to do with the trade network and players should have their DP memorized anyway. This means only resource deals and gold-per-turn gifts (rare) are listed. Resource deals require a trade connection, so it makes sense to show them in this context.	
The hover text for the food display on the city screen shows a simpler breakdown when food only comes from worked tiles and is lost only through consumption.		BtS shows no hover text on the food display. BULL shows a very through breakdown with several subtotals. A BULL comment (copied into AdvCiv) above `CvGameTextMgr::setFoodHelp` (cpp file) shows the format of the breakdown.
Rationale	Food production and loss in most cities are very simple; no need for any subheadings.	

088	Keyboard shortcut for unselecting all units
See also	002: other cosmetic changes 004: other misc. UI changes

	Part of [154](#) (unit cycling button) is implemented based on this change.

AdvCiv	BtS
Can unselect all units by briefly holding Alt+Shift+U.	The only way to select no unit is, to my knowledge, to give orders to all units. While a unit is selected, an animated circle is shown around it on the main map and the unit action buttons are displayed.
Rationale	For just studying the map for a while or for taking a screenshot. The fewer distractions, the better.
	Alt+U is already taken by the Unit layer; Shift+U by "unload all". Unfortunately, the unit debug menu blocks both Ctrl+U and Ctrl+Shift+U.
	"Deselect" seems to be the more commonly used (and, in a way, more linguistically correct) term, but I associate that more with unchecking an option.
Config	Through `XML\Units\Civ4ControlInfos.xml`. The key combination can't easily be changed there however because I've hardcoded the same combination in `CvGlobals::suppressCycling`. If automatic unit cycling is not suppressed, then it'll immediately select another group (or the same as before). That's also why the keys need to be held for a moment.

089	Don't show odds for impossible attacks	
AdvCiv		BtS
The UI doesn't show combat odds for illegal attacks. Restructured `CvPlot::getBestDefender` and its auxililary functions a bit so that the code that checks whether a defender is valid is more clearly separated from the code that selects the defender with the highest priority.		Through the Alt key, odds can be shown e.g. for attacking one's own units or units that are already maximally damaged.
Tbd.	In the [defender randomization](#) branch, the distinction between defender selection and validity is even clearer; merge that perhaps.	
When hovering for attack and no legal attack is possible and some attacker would be able to attack if its damage limit wasn't reached, then help text says that the defenders are maximally damaged and states the attacker's damage limit.		No help text about damage limits; just a red cursor to indicate that the attack move isn't allowed. The UI generally doesn't explain why moves aren't allowed.
Rationale	It can sometimes take players a moment to realize that the damage limit has been reached as opposed to e.g. the selected unit having no moves left. One player has also (mis-)reported a case of maximally damaged defenders to me as a potential bug. It's also nice to see the damage limit value (percentage) right away.	

090	Field of view (FoV) slider	
See also	See [004m](#) for an explanation of the FoV value. 004m sets the default FoV value and camera start distance. [CFC post](#) (the quoted part) criticizing the slider.	
AdvCiv		BUG

The length of the FoV slider is set based on the screen dimensions. The text labels go from 10 to 100%. The percentages below 75 correspond one-to-one to (BtS) FoV values; above 75%, each percentage point increases the FoV value by 2, meaning that 100% corresponds to 125.	The length is hardcoded to 100 and the labels go from 0 to 100. That is, the slider position corresponds directly to the FoV value.
Rationale	A longer slider makes it a bit easier to position the slider at a particular percentage. That said, there seems to be a bug in the slider component (in the EXE) that makes the slider position change by 1 when releasing the mouse button. A much longer slider might (mostly) fix that problem, but some players keep the slider permanently on display, so it mustn't take up much space. FoV values below 10 aren't useful for anything, so I've removed those from the slider. At a value of 100, Huge maps still don't fit entirely on the screen without zooming into Globe view, but 125 seems to be enough. The high slider positions aren't useful for playing the game, only for screenshots and AI Auto Play. For the low and medium slider positions, I want to keep the 1:1 relation between slider percentage and actual FoV value so that players who have their favorite FoV value memorized don't need to adjust.
Tbd.	I've tried to work around the slider precision problem in `CvMainInterface.handleInput`. It might work if one could distinguish between the user hovering over the slider and dragging the slider; but I see no practical way to do that. Anyway, I've left some comments in the code. I think PlatyUI uses +/- buttons instead of a slider. That's probably the better solution.

091	Changes to score graph (Info screen)
See also	[004s](): Yield graphs show moving average [077](): Changes to Demographics tab
Credits	[This]() post (last third) by crullerdonut made me aware that scores are generally visible without the need for espionage.
AdvCiv	*BtS*
Once the demographics of another civ have been visible to a human player, the score graph remains visible for the rest of the game regardless of espionage. When the human player has never seen another civ's demographics, the score graph is shown for turn numbers greater than or equal to the turn on which the civ was met by the human. Exceptions: • No graph is shown for civs that were only met during the 5 most recent turns. • When loading a savegame from AdvCiv 0.97 or earlier, civs are assumed to have been met on the turn that the savegame was created on, and no prior espionage visibility is assumed. (The respective data hadn't been stored prior to AdvCiv 0.98.)	The score graph shows either the full score history of a civ or none at all. Seeing the full score history requires the ability to see the target's demographics through espionage. The current game score, however, is visible on the scoreboard for every known civ.

I've implemented this behavior for both the BtS and the BUG Graphs tab.	
Config	The part about partial score graphs is optional and, by default, disabled. The option is named "Partial Score Graphs" and located on the Advisors tab of the BUG menu.
Rationale	Want to make the score history available if the player has been able to observe it. Don't want to reveal the entire score history on the first meeting though. The 5-turn restriction is supposed to avoid a very short visible interval that would be barely noticeable. Partial graphs disabled by default because they're not useful enough to risk confusing players.
See also	Screenshot of some partial score graphs attached to this CFC post

092	Larger minimap on the main interface – **incomplete**. Hardly anything done so far, and no visible change.
See also	If the minimap is enlarged, the unit pane on the lower left should be enlarged as well. That'll probably also enlarge the help text area, making help text fit more comfortable despite the enlarged font (002b). Kjotleik has enlarged the minimap without touching the unit pane: CFC post (with screenshot; code attached)
Credits	VDNKh had suggested to increase the minimap size (CFC post).
Rationale	HD displays

AdvCiv		*BtS*
On HD displays, I'd like to increase the size of the pane on the lower right by about 20%, pushing the scoreboard up a bit and decreasing the width of the unit action pane on the bottom. The pane on the lower left should be enlarged as well for symmetry.		No part of the main interface "HUD" is adjusted to the display resolution.
Tbd.	Hardly anything done so far, just some variables in `CvMainInterface.py`. No visible change yet.	
Rationale	Vertical space is still not abundant on HD displays; width is abundant. The unit action pane is much wider than it needs to be. A 20% size increase looks good to me in a mockup.	
In addition to the panes and minimap, the city screen could also use wider columns on the left and right. The columns should extend beyond the width of the panes on the lower left and right.		On widescreen displays, the city screen uses all the additional width for the main map background in the center.
Tbd.	Nothing done yet.	
Rationale	The background area is only needed for showing worked tiles. It doesn't need any width beyond square dimensions.	

093	*Still unassigned*

094	Production decay help merged from BULL
Config	Can be enabled on the City tab of the BUG menu. No separate options for the hover text, i.e. the hover text will always be shown along with the warning indicators (exclamation marks). Disabled by default – however, when upgrading from earlier versions of AdvCiv (i.e. pre-v1.0), the option will be enabled because it had already been present (hidden) and had been set to enabled by default (as in BULL).
See also	Requested [here](#) (2nd quote box)
Tbd.	The red hover text is kind of difficult to read. And it would be nice if its color could match the color of the warning indicator (yellow exclamation mark if decay isn't imminent). I've tried yellow, and coloring only some words; didn't like that either. Not going to invest more effort for now. Not sure if production decay could simply be disabled and this option removed. In theory, players could keep a small army in their production queue to avoid unit expenses and (in multiplayer I guess) to hide their strength.

095	Optional wide city bars
Config	Can enable the wide city bars from BUG on the "Map" tab of the BUG menu; disabled by default. BUG had implemented the wide bars entirely through custom graphics; K-Mod had removed those graphics. The AdvCiv option loads the custom graphic dynamically. Note that this mechanism will break if the custom graphics (`Art\Interface\CityBar`) are renamed or if the path to the original graphics (in `XML\Art\CIV4ArtDefines_Interface.xml`) is changed. I've tried unsuccessfully to place the custom graphics in a folder `CityBarWide`, which would've been a bit more robust. Updating the paths in the customized `.nif` file wasn't sufficient to make it work.
See also	Related (short) CFC [post](#) (near the middle)

096	Tech tree reflects the current game state (to an extent)
Tech help text on the tech tree takes into account the active player's context for unknown techs. Will e.g. not list the free Great Artist from Music if another civ has already discovered Music. This change does not hide any ability icons in the tech boxes.	
See also	I've posted about this on CFC [here](#) and [here](#) (items 1 and 2).
~~Show tech-religion icons on the tech tree.~~ (Don't do this after all. I've commented the code out.)	Tech-religion icons (tiny religion icon superimposed on the icon of the tech that founds the religion) are shown in choose-tech popups and on the main interface (when no research is chosen), but not on the tech tree.
Rationale	I wanted to show the icons so that players can tell more easily which techs will still grant a religion when playing with the Choose Religions option. However, the tech-religion icons are difficult to make out on the tech tree (smaller icons there than on the main interface) and, currently, the whole tech tree has to be redrawn when an icon has changed since the last time that the tech tree was opened, and this results in a noticeable delay. Not quite worth it I think. One can still tell through hover text on the tech tree (see above) whether a tech will found a religion.

097	*Still unassigned*

098	Changes to culture spread	
AdvCiv		*BtS/ K-Mod*
A city can spread tile culture onto another landmass only at a range of one tile beyond the city's culture level. Within the same landmass the range is equal to the culture level plus 3 as in K-Mod. On water, culture can spread only onto workable tiles as in K-Mod and BtS.		In BtS, cities spread tile culture only within a radius equal to their culture level, i.e. to tiles within the city's culture range. K-Mod has extended the tile spread range by 3 tiles beyond the (proper) culture range in order to allow a higher number of cities to participate in culture "wars" over contested tiles. In both BtS and K-Mod, a city can spread culture only to tiles that are within the workable range (2 tiles) from some tile on the same landmass as the city. This means, tiles within the culture range of a city can remain neutral because tile ownership requires at least 1 point of tile culture.

Rationale	To prevent puzzling situations like this (A, B - cities of the same owner, L - a non-city land tile, W - a water tile):
	
	`ALL`
	`WWW`
	`WWW`
	`LL`**`L`**`WLLB`
	
	None of the land tiles on the bottom are workable from the area of A. The bolded land tile is workable from the area of B and, under the K-Mod rule, B only needs culture level 1 ("poor", 0-9 city culture) in order to spread culture to that land tile. The result is an owned tile that has only one adjacent owned tile (the water tile to the northwest) and that has two unowned tiles between itself and the borders around B.
	
	Under the AdvCiv rule, the bolded land tile will be unowned.	

See also	Doing away with the "only spread onto workable water tiles" rule could make surprise attacks against large coastal cities more difficult (see 162). That said, having ownership of water tiles that are positively non-workable could be confusing, and I think the BtS rule reflects international law pretty well. 099f prevents culture spread onto unowned tiles. 025: Reduced culture spread from vassal to master	
Increased the base rate of culture spread ("free city culture") a little bit (from 4 in K-Mod to 5), making culture spread from buildings a little less important.		
Config	I've put the `CITY_FREE_CULTURE_GROWTH_FACTOR` global define (`GlobalDefines_advc.xml`) back in use, albeit with different semantics than in BtS because culture spread works very differently in K-Mod.	
Rationale	One measure for preventing culture rates from buildings, especially wonders, from being overwhelming in the early game – now that I've reverted most of the K-Mod changes to building culture.	
See also	200, 201: Changes to building culture rates. 099b: Tile culture decay – and expedited decay on "stolen" workable tiles – should also (and probably more so than the base culture rate tweak) help rein in culture spread from wonders.	

AdvCiv	BtS
Once a wonder is obsolete, its culture rate doubles. Building culture rates do not double after 1000 years. Disabled by default; instead added a loading screen hint about the BtS rule.	1000 years after a building has been constructed, its culture rate gets doubled. This rule is, as far as I can tell, entirely undocumented. The UI also doesn't show the construction dates anywhere.

Rationale	Disabled this again after a quick test because this change reduces the contribution that wonders can make toward a Culture victory too much – wonders that never go obsolete, that is. Would have to accompany this change with large culture increases for Medieval wonders and would have to set obsoletion techs for most wonders.
	The original idea was to make the rule less obscure and more transparent, and to make an early commitment to a Culture victory less decisive.
Config	Can be enabled in `GlobalDefines_advc.xml` through `DOUBLE_OBSOLETE_BUILDING_COMMERCE`.
See also	Some other disabled rule changes are collected under change id 500.

No culture spread from cities during anarchy.	Occupation sets a city to "no culture level", which also blocks culture spread. I don't think anarchy prevents culture spread.
Rationale	Seems more consistent with the idea of anarchy halting economies. E.g. Cottages also don't grow during anarchy.

099	Culture isn't removed when a civ is destroyed
See also	Could merge kekm.24 fully in order to stop colonial vassals from using the slot of a dead civ (which, I suppose, resets the culture of the dead civ).

AdvCiv	BtS
When a civ is eliminated, its culture remains in the game and keeps causing anger.	When a civ is eliminated, its culture disappears as well.
The AI player is announced as having been "defeated".	An announcement says the *civ* was "destroyed".
Anger before and after conquest is explained as "We resent being ruled by a foreign culture".	Before elimination, anger from culture is explained as "We yearn to join our motherland". No anger after elimination (the motherland is gone).
Anger is 40% of the foreign population with Open Borders (or capitulated vassal or Barbarian), 125% times 40% = 50% without OB (or if they're defeated), and 175% times 40% = 70% while at war; all rounded down.	Anger is 40% of the foreign population while at peace, and 150% of that, i.e. 60%, while at war, both rounded down.
(Until v0.96, the multipier for no Open Borders was 150%; reduced in response to this post by xyx.)	
Rationale	Removes one strong incentive for conquering a civ entirely. The goal is to make a vassal agreement the correct choice most of the time – less micro-management that way. Also intended to weaken military strategies.
	It's also glaringly implausible that a culture would instantly disappear.
	No AI changes: conquering cities is still very desirable, and the AI doesn't consider anger from culture anyway when deciding whether to accept a capitulation.

Config	The anger can be tweaked through XML (`GlobalDefines_advc`), but can't re-enable cultural elimination. This is because BtS doesn't actually set culture to 0 in one place, but checks for elimination in numerous places and treats culture as 0 when appropriate, so making it optional is messy.
Credits	Contributions by Chronis, lfgr and vincentz CFC thread
See also	Chapter about "Immortal Culture" 130w adds a diplo penalty for cities with high foreign culture.

099b	Tile culture decays

Each round, the culture value of each player (civs and Barbarians) in each tile is decreased by 1.3%. When a tile is within the city radius of at least one civ, the tile culture of civs that don't have the tile within a city radius decays faster: if the tile is within the outer ring of any city radius, 2.3 percentage points are added to the decay rate, and 4.6 percentage points if it's within the inner ring of any city radius. Exception: civs with less culture in the tile than the city owner with the highest culture don't suffer from expedited decay. (This change does not affect *city* culture.)	Tile culture doesn't decay. (But it's treated as 0 when a civ is eliminated.)

Rationale	Should lead to fewer unworkable tiles (see also the rationale for 035) – especially with K-Mod's increased range of culture spread in mind – and fewer issues with stranded AI stacks (see also rationale for 034). Higher decay rates would accomplish more, but I don't want tiles to flip too easily around conquered cities. That said, in BtS, it's too difficult to culture-flip tiles in the radius of a conquered city, especially in Renaissance or later; tiles have tens of thousands of culture points by then. It generally takes too long for city culture rate to translate into relative tile culture in BtS. Why not simply tweak the culture spread formula so that tiles near cities receive more culture? The player only sees the culture percentages, so, visually, it doesn't make a difference. I don't want to increase culture on tiles that are near multiple cities of the same civ lest that "*two cities would almost always be able to culture press a solo city*" (from the K-Mod readme file). The main goal is to allow civs to work tiles in their (exclusive) city radii. That's also why I'm exempting civs with less tile culture than the city owner. It's fine if another civ has 45% tile culture so long as the 55% civ is able to work the tile. The exemption also makes it easier to implement this change without a performance penalty.
Config	`TILE_CULTURE_DECAY_PER_MILL` and `CITY_RADIUS_DECAY` in `GlobalDefines_advc.xml`
See also	201 restores most of the building culture rates that K-Mod had reduced. This change was introduced with AdvCiv 1.0. At the same time I've increased the decay rates by 0.3 percentage points (originally, I had used 1%/2%/4%; adding half a percentage point to that seemed too much in a test game) and one of several tweaks to compensate for the building culture changes. Moreover, I had had the impression that mature conquered cities were still taking a very long time to acculturate, and this CFC post also argues (toward the end) that the decay is too slow. 035 (disabled): City owns all tiles that no other city could work It turns out that "RoM: A New Dawn" also has a culture decay mechanism

	(SourceForge link to revision), but theirs is only triggered every 50 turns and then halves tile culture. For performance reasons I suppose; but I've managed to make the computing time negligibly short in my implementation after some tweaking.
Cities in occupation spread no tile culture.	Cities in occupation produce 0 city culture, but the present city culture generates tile culture as normal.
Rationale	This should give civs that allow their cities to revolt a harder time.

No increased decay in city tiles, nor in tiles owned by Barbarians. And increased impact of stolen tiles in the inner radius of Barbarian cities on revolt chance.	
Rationale	Don't care if Barbarians are able to work tiles. Rather give those tiles to civs with strong culture. Also increase the revolt chance then so that engulfed Barbarian cities get put out of their misery before long.
	Culture in city tiles doesn't affect which tiles are workable. The increased decay would make revolts and anger from foreign culture easier to handle, which isn't what I want. The exception for city tiles can lead to higher culture percentages of the city owner in tiles surrounding the city than in the city tile. This is a bit odd, but not a real problem I hope.
099c	Cities revolt regardless of culture range
See also	210b adds an alert about positive revolt probabilities

Any city with sufficiently strong foreign culture can revolt, but the revolt can only flip the city if the owner of the foreign culture is alive and has a city nearby. (And can't flip if flipping is disabled in game options.) Made some adjustments to AI willingness to accept capitulation vs. trying to capture further cities.	Foreign culture can only cause a city to revolt if the owner of that culture is alive and has a city nearby, i.e. close enough so that the first city is within the culture range (based on culture level) of the second city.
Rationale	To remove another incentive for eliminating civs, to make conquests more costly and culture more relevant.
	Resurrecting a prior owner seems needlessly messy; the probabilistic occupation times (change 023) should make revolts from dead culture painful enough. Joining the Barbarian civ would be easy – but strange, I think, because the city would then train units based on the Barbarian tech level.
	Not sure if cities with high living foreign culture but outside of foreign culture range should be allowed to flip. Let's say the rebels only dare joining the homeland if the homeland is near enough to protect them.
Config	Can (largely) restore BtS behavior through `REVOLTS_IGNORE_CULTURE_RANGE` in `GlobalDefines_advc.xml`.

Barbarian culture can cause cities to revolt. They can't flip to Barbarian control though, and Barbarian culture strength is only counted half in surrounding tiles. Cities can't flip from a master to its vassal; only go into occupation. Increased strength of foreign culture in cities of capitulated vassals.	Essentially the same, but Barbarian cities have practically never enough culture range to cause another city to revolt. Culture in tiles adjacent to the city always contributes to the foreign culture strength, and, ultimately, the revolt chance. Can flip between vassal and master. No way then for the master to get the city back.
Rationale	Not sure about this. Seems more flavorful/ historical to let conquered Barbarians

	revolt. Might play better without these revolts; players don't expect them, and conquering Barbarians should be easier than conquering civs. Then again, Barbarians don't generate much culture, so it doesn't take much effort to suppress them, at least not when the surrounding tiles don't count. Could say that only counting the city tile models the (political) disunity of the Barbarians. Could easily implement flipping to Barbarian control. Historically, that hasn't really happened; see the list of colonial uprisings [here](#) (Wikipedia). More accurately modeled as a period of unrest. Flipping from master to vassal could incentivize elimination over capitulation. Also, I find it ultimately more annoying than challenging.
Config	`BARBS_REVOLT` switch in `GlobalDefines_advc.xml`
See also	[025](#) reduces culture spread from capitulated vassals

If a city can't flip on the third revolt (because the cultural owner is dead or flipping disabled in options), it loses one population instead (and goes into occupation).	No extra penalty if flipping isn't possible; occupation periods just keep getting longer.
Rationale	To discourage players from ignoring revolt probabilities (the lost turns don't become punishing until numerous revolts have taken place). And it would be strange if cities could just keep revolting. The loss of population reduces the revolt probability so that the city should eventually end up depopulated but pacified.
Extra AI defenders in cities that might revolt. Some of these come from the floating defender pool. The AI also trains some more floating defenders when struggling with foreign culture. When too many defenders would be necessary (based on city population and era), the AI will not assign any extra defenders and essentially wait for the city to flip.	No code for this at all. The AI tends to put units in border cities though, against external threats.
Rationale	Since revolts no longer occur only in border cities, the AI had to be adjusted.
See also	[advc.ctr](#): The AI for city trades might cede cities under foreign culture pressure. (Not yet sure how that will work.)

099d	Can't spread religions under occupation.
AdvCiv	*BtS*
Missionaries and Executives can't spread religions/ corporations in cities that are under occupation.	Can spread regardless of occupation, but there isn't much of a point because happiness and culture from religion have no effect under occupation, and revolt chance is 0.
Rationale	Now that the occupation duration is linked to the revolt probability, spreading a religion right after conquering a city could reduce the (expected) occupation time. Don't want players to micromanage missionaries to spread religions right after conquest. Also seems realistic that religious communities (or corporate branches) can't be established during disorder. (But don't want to apply this rule to disorder from anarchy because anarchy doesn't normally restrict the actions of units.)
Tbd.	Would be better to show the Spread Religion button grayed out with explanatory text. Currently, the button isn't shown when a city is in unrest.

099e	AI for promotions that reduce revolt chance	
AdvCiv		*Warlords*
The AI ignores revolt protection when selecting promotions and when moving units.		A promotion with revolt protection is chosen based on whether the cultural owner of the current tile differs from the actual owner.
		When moving units, revolt protection is ignored.
		Revolt protection from promotions was added in Warlords but remained unused. K-Mod gave the Leadership promotion 50% revolt protection.
Rationale	Too much work to do this properly (see *Tbd.* below). What Warlords does is worse than nothing.	
Tbd.	The promotion should be selected based on the revolt chance in a nearby city, the number of units already stationed there and the revolt protection percentage. When there is a city with a high revolt chance, a unit with revolt protection should be given a defensive Unit AI type and stationed in that city.	

099f	Culture on unowned tiles	
AdvCiv		*K-Mod*
Culture can't spread to unowned tiles. (I.e. cities still spread culture beyond their own ownership radius, but only to tiles that are within the ownership radius of another city of the same or of a different player.) Culture percentages are shown when hovering over any actively visible tile. The Culture layer still only colors owned tiles.		K-Mod increases the radius of culture spread around cities beyond the radius of cultural ownership, meaning that it's common for unowned tiles to have tile culture of one or multiple players. As in BtS, culture percentages are only displayed on owned tiles (help text, Culture layer). In BtS, the only way for unowned tiles to have positive culture values is through city razing.
Rationale	Invisible culture is not a good thing; can in particular be an unpleasant surprise when founding a city near a foreign border. Showing culture values on unowned tiles is distracting when most of the visible unowned tiles have culture values. (But it's OK when culture on unowned tiles occurs rarely, i.e. only after razing.)	
	I also don't think that culture spread to unowned tiles serves an important gameplay purpose. Yes, founding right next to a foreign border should be a struggle, but it's still difficult enough if the foreign culture starts spreading only once the city has been founded; in fact, that's probably better for gameplay when it comes to cities founded in the midgame and late game. The main purpose of the increased culture radius is to let non-border cities with high culture output support border cities in their struggle for contested tiles; the point isn't to spread culture to tiles that aren't really near any city.	
	I think it's nice if border cities can also spread culture fairly deep into foreign territory and thereby pave the way for assimilating that territory after a war of conquest; therefore, I'm not limiting culture spread to tiles within the ownership radius of a *friendly* city.	
	Culture layer: While it would make sense for a layer called "culture display" to prioritize info about culture over info about ownership, there has to be some way to see tile ownership in Globe view, and, currently, the culture layer is the only way (cf.	

	[004z](#)).
Config	Until AdvCiv 1.0, there was a BUG option for showing culture on unowned tiles and it was always possible to hold down the Shift key while hovering over a visible tile in order to show the culture percentage. All that code still exists and is tagged with "advc.099f" – it's only commented out.

099g	Order of tile culture percentages in help text	
AdvCiv		*BtS*
Show the tile owner's percentage on top, then the percentage of the active player and the rest in descending order.		Tile owner on top, the rest in turn order.
Rationale	Not sure if ordering by culture is really an improvement. I don't think it hurts as lines don't align anyway when moving the mouse from one tile to another; so a uniform order doesn't really help. Showing the active player second (or first if it's the tile owner) should be an improvement in scenarios when the active player isn't in slot 0.	
See also	I also wanted to try putting all percentages on a single line – to make more room for the unit list ([061](#)). However, the single line seemed quite a bit harder to read or would at least take some getting used to. So the code for that remains commented out in `CvGameTextMgr::setPlotHelp`.	

100	Changes to sponsored wars	
See also	[146](#): Peace treaty between sponsor and the civ that declares war. [210a](#): Alert about AI willingness to be hired for war.	
AdvCiv		*BtS*
When an AI civ declares war at the request of another civ (hired/bribed for war), the DoW message informs the human players about this sponsorship: "...declared war on ... at the request of Alexander."		Humans don't learn about AI bribes.
Rationale	The information isn't terribly relevant strategically, but makes it easier to make sense of the AI, and makes it clear that the AI actually does sponsor wars. Moreover, if AI civs learn about sponsored wars ("You brought in a war ally against us!"), the human civs should be allowed to know this too.	
(Only relevant if UWAI is disabled:) AI civs are reluctant to be hired for war against a powerful civ. Either they decline fearing "their military might", or the price for war is high, up to two times higher than in K-Mod, depending on how disparate the power ratio is.		Weak AI civs are sometimes bribed into joining wars against powerful civs. An AI civ doesn't consider its individual military power but only the total power of the war coalitions.
Rationale	War against a considerably stronger civ is a big risk, even as part of a coalition, and, in any case, the weakest link is unlikely to conquer any cities. Moreover, sponsoring wars was too cheap overall.	
Config	`WAR_TRADEVAL_POWER_WEIGHT` in `GlobalDefines_advc`.	

While negotiating peace, no declarations of war against third parties can be discussed: no "Declare War On" items are shown on the trade screen.	Normally blocked by "we don't like you enough" or "will never trade with our worst enemy," but it may occasionally be possible to hire an AI civ for war as part of a peace treaty.
Rationale	Shouldn't show the option if the AI practically always refuses, and Cautious attitude despite war is an oddity that shouldn't matter. Could instead remove the attitude checks while at war. Seems more realistic that such options can't be discussed. Would also take a bit of implementation work because the the war trade should be evaluated assuming that the peace treaty is already signed.
100b	Brokered peace shown in replays
Brokered peace is shown in the replay log ("brokered by ...").	Replay log only says "has made peace with".

101	Revolt after conquest
See also	[210b](#) shows an alert when revolt probability changes from 0 to a positive value. [ctr](#) shows positive revolt probabilities on the Comestic Advisor.

AdvCiv	*K-Mod*
Reverted the K-Mod 0.9 change to revolt probabilities, so that foreign culture strength approaches 100% as the city owner's tile culture approaches 0% (as in BtS). Foreign culture strength is taken to the power of 1.35 and times a (player) difficulty-based factor between 0.1 (Settler) and 0.4 (Deity). The progression gets flatter toward Deity, e.g. reaches 0.25 at Noble and 0.35 at Emperor.	Foreign culture strength goes to infinity as the city owner's tile culture approaches 0%. As a consequence, sometimes no stack is large enough to keep a city from revolting, or at least not large enough to reduce the revolt probability to 0. On the other hand, a medium-size stack is often enough to reduce the revolt probability nearly to 0 but not all the way. The difficulty level does not affect revolts.
Rationale	The K-Mod behavior is intended, i.e. it's not that karadoc hadn't considered flipping after conquest; see discussion [here](#). I can see how keeping a city can be harder than conquering it in the first place – but not *that* much harder. Say, ten good units should generally suffice to hold on to a city. If this makes revolts easy to prevent late in the game – fine; it's mostly early expansion that needs to be paced.
	The exponentiation is supposed to balance out changes to the culture garrison strength formula (see below). The player handicap wasn't added until AdvCiv 1.0; before that the Emperor factor had applied regardless of the difficulty level. Perhaps should have thought of that earlier. On lower difficulty settings, smaller stacks can conquer cities, so smaller stacks should be able to hold onto them. That being said, I think it has been mostly Emperor players and above who have complained about revolts being too big a factor. I guess they're the ones used to conquering numerous cities in quick succession. I think it's a fairly moderate effect, at least above Noble.
	The handicap change also stacks the game a bit further in favor of the AI, which will e.g. need only 83% as many garrisons as a human player on Emperor difficulty. Seeing that I haven't written much code for helping the AI avoid revolts better, I guess that's for the better. Could easily be changed so that culture strength gets based on the game handicap (shared by all players).

Config	The exponent and normalization factor can be adjusted through `GlobalDefines_advc.xml` (`FOREIGN_CULTURE_STRENGTH_EXPONENT`, `FOREIGN_CULTURE_STRENGTH_FACTOR`). The factor can also be used to adjust how difficult it is to suppress revolts overall.

AdvCiv	BtS
The culture garrison strength (CGS) of a unit equals its current combat strength with the following modifiers: • City defense modifiers (Archery units, Garrison promotions) • Generic strength modifiers (Combat promotions) • -50% for non-lethal units (Siege units except Machine Gun) The `iCultureGarrison` values in XML are ignored except that `iCultureGarrison=0` in XML still means 0 CGS. All non-land units have 0 CGS (as in BtS). Hover text on the main map and city screen (nationality bar) says how much more garrison strength is needed to bring the revolt chance down to 0, or, if it's already 0, how many units could be safely relocated.	CGS values are set in XML (`iCultureGarrison`) to e.g. 3 for Warrior, 4 for Archer, 6 for Maceman, 9 for Rifleman, 12 fopr Tank. The total garrison strength of a city is the sum of these values. E.g. Catapult has `iCultureGarrison` 4, same as Axeman; Machine Gun 9, same as Rifleman. Hovering on the nationality bar shows the revolt probability; other than that, no help text.

See also	023 reduces the CGS of damaged units. 500b (disabled by default) introduces a defensive strength (DS) value that determines whether citizens fear for their safety. It seems that using the exact same formula for CGS and DS wouldn't work well. In particular, while some effect of Walls and Castle on CGS is plausible, they shouldn't be a major factor (cf. this CFC post of mine).
Rationale	It's mainly about the help text. Players will often want to station exactly as many units as necessary to bring the revolt probability down to 0. By making CGS nearly equal to combat strength, players don't need to be aware of the values set in XML. The BtS XML values also increase too slowly over the course of the game. This makes outdated units too useful for revolt suppression. Combat strength increases too steeply throughout the game compared with foreign culture strength. I'm balancing that out through a change to the foreign culture strength formula (see above). That formula is now very complicated (see also below; referred to as "revolt strength" there), but it was already fairly complicated in BtS, and the crucial thing is that CGS has a simple, predictable effect so that players know how many units they need in each city. As for the Siege modifier, siege units just seem badly suited for quelling revolts. It's asymmetrical warfare.
Credits	Elkad and VDNKh have asked for help text to be added (CFC link).

Tbd.	Currently, the main map doesn't show how much CGS is needed while a city is in occupation; perhaps that should be changed.
	Should perhaps explicitly show the effect of revolt protection (Leadership promotion) in the hover text of the nationality bar. Perhaps wait with this until I add revolt protection to Jail.
	Armored units having *the* highest CGS values is perhaps a bit counterintuitive (and makes them even more powerful). There are historical examples of tanks being effective at suppressing revolt though ...
	Consider giving Spy a positive CGS, possibly also recon units. Espionage should somehow help against revolts. ("Spread Culture" does in K-Mod, but needs to happen before conquering the city, which requires some foresight.)
Config	I've written code for a partial breakdown of the revolt probability but didn't find it helpful enough in the end. The code is commented out in `CvDLLWidgetData::parseNationalityHelp`.

Revolt chance is adjusted to game speed through the Golden Age modifier (e.g. revolt chance divided by 125% on Epic speed).	Through the victory delay modifier; e.g. divided by 150% on Epic speed. BtS didn't have any speed adjustment.

Rationale	Culture spread is arguably the most important factor for revolt supression, and game speed has no impact on culture spread.

Foreign culture strength increases with the ratio of techs known to the cultural owner (even if that player has been defeated). Over time, that ratio shifts toward the maximum of the owner's and cultural owner's known-tech ratio. The ratio gets quantized to a number between 0 and 10.	Increases with the game era. And the game era only affects the culture strength counted for surrounding tiles, not the culture strength from city population.

Rationale	This era modifier is pretty noticeable, so plausibility matters. Involvement of third parties (game era) isn't really plausible: if the city owner hasn't been able to acquire their tech, why should the insurgents be able to. The one civ that would be likely to deliberately support the insurgents is the cultural owner itself.
	Using the tech era of the city owner is undesirable because it could discourage that player from reaching a new era. Era numbers are also a bit too coarse. However, a smooth increase of revolt probabilities would be even worse because it would encourage players to make slight changes to city garrisons all the time.
	Using only the tech of the cultural owner would encourage the city owner to defeat or stifle the cultural owner. A slight incentive for that is probably a good thing, but I don't want it to matter much most of the time.

See also	[CFC post](#) (last spoiler box) observing that foreign culture strength can't keep up with late-game garrisons. That was when the tech of the cutural owner had only affected the culture strength counted for surrounding tiles (as in BtS). And I#ve made some other tweaks in response to that post.

AdvCiv	BtS
Revolt strength is reduced based on the highest city tile culture of any civ, even if that civ doesn't have a nearby city. For example, when the owner of a city has almost no culture, say, 1%, and the only other nearby civ has some culture but also not much, say 10%, the revolt probability is low or zero.	Only tile culture of nearby civs matters. The 10% in the example would lead to a high revolt probability because it's much more than the owner's 1%.

Rationale	Don't want to reward civs for just 10% (or so) culture, especially not uninvolved parties in wars. Also implausible that the 10% would find sufficient support for a revolt.	
See also	A somewhat common issue because of [099](#) (culture of dead civs sticks around).	
Revolt strength reduced if foreign tile culture isn't far higher than owner's tile culture. E.g. if foreign tile culture is just 25% greater than owner's culture, revolt strength is reduced to 25%. This makes revolts easy to suppress with one or a couple of units when the owner's tile culture is close to the foreign tile culture.		Revolt strength is increased by up to 100% if owner's tile culture is small, but high owner's culture can't reduce the base strength from population and surrounding tiles. Cities with up to 49% tile culture of the owner can still have a high revolt chance.
Rationale	Counterintuitive that a city could easily flip at nearly 50-50 tile culture. The change rewards the owner for building up culture, even if the owner hasn't (yet) reached parity.	
City flipping after conquest enabled by default. (Actually, flipped the option to "No city flipping after conquest" and kept it disabled by default.)		The option is called "City flipping after conquest" and disabled by default.
Rationale	Part of my efforts to weaken military strategies. I want to build on the revolt mechanism, so it should be (fully) enabled by default.	
When a city flips while at war, the garrison is bumped (moved to the nearest legal tile). Only Barbarian garrisons are killed.		The garrison is killed. Bumping only happens when flipping at peace-time.
Rationale	Killing a stack of units is pretty outlandish. Bumping Barbarians would be unusual, and killing them is less problematic wrt. game balance. I guess they just lay down their arms.	
Credits	The issue had been pointed out (and narrowed down) by DarkLunaPhantom [here](#).	
Added the number of prior revolts to the Nationality help text, and whether the city will flip on the next revolt.		Help text only shows the revolt probability.
All civs that know a city are notified about a revolt.		Only the actual owner and the cultural owner are notified.
Revolt probability (if > 0) shown on the main map as part of the city tile help text.		Need to enter city screen to see revolt probability; no way to see revolt probability of a foreign city.
Config	Option on the BUG menu (Map tab) to show the revolt probability in the city bar hover text instead.	
See also	Help text for air unit capacity ([187](#)) is similar to revolt probability insofar that BUG show that information in the city bar hover text (and AdvCiv moves it to the tile hover text by default). [advc.ctr](#) shows revolt probabilities on the (non-BUG) Domestic Advisor screen. MNAI shows a projection of the number of turns until 50% nationality is reached in help text ([Git commit](#)). Might be helpful(?); not so easy to compute in AdvCiv because of tile culture decay ([099b](#)).	

Rationale	Revolt probability in visible foreign cities shouldn't be secret – all factors are public knowledge. City-related information normally belong into the city bar hover text – the tile hover text doesn't even say whether a city exists in the tile. However, for deciding whether culture garrisons can be moved out of the city, it's more convenient to have the revolt info in the tile hover text along with unit info. One can argue that the revolt info isn't strictly city-related because it includes info about culture garrison strength, which is unit-related.

A fist icon can be shown above the billboards of cities where the revolt probability is positive. The icon is then still shown (along with a countdown) above the billboards of cities under occupation, even if those cities have a revolt probability of 0.	The fist icon is shown above the billboards of cities under occupation, along with an occupation countdown. No billboard icon for positive revolt probability.

Config	Option on the BUG menu (Map tab); disabled by default.
See also	002f deals with other billboard indicators.
Rationale	Seems useful, but clashes with the occupation indicator, and I can't think of a different – self-explanatory – icon for positive revolt chance. A yellow fist would make sense, but the icon is very tiny, so a different color will probably make it difficult to recognize as a familiar icon.

Revolt probability is initially based on the current population of the city. If the city has ever had a higher population than currently, then, over the course of 37 turns (slightly longer or shorter on speed settings other than Normal), a gradually higher population is assumed, up to a maximum of the highest population ever. Then the assumed population decreases again toward the actual population.	Always based on the highest-ever population of the city.
Culture in the eight adjacent tiles factors into foreign culture strength; the effect increases over time (same mechanism as above for population). After 50 turns (on Normal speed), culture strength is further increased when adjacent tiles are *owned* by the foreign civ.	Ownership of the adjacent tiles always counts, and can increase culture strength significantly. Culture points in those tiles don't count directly.
Each citizen angry about sacrificed population ("we cannot forget your cruel oppression"), is counted as 2.5 population (instead of just 1) and the resulting foreign culture strength is increased by a modifier based on the square root of the oppression anger and speed-adjusted; e.g. 50% for one angry citizen on Normal speed. The AI doesn't use Slavery in cities that already have a positive revolt chance.	Sacrificing population doesn't lead to a lower revolt chance because only the highest-ever population matters for that. It can get rid of 2 citizens angry about foreign culture for the price of 1 being angry about oppression.

Rationale	I think the BtS rules are supposed to make cities flip faster once the adjacent tiles have flipped. Loss of tiles will often lead to population loss, which would normally reduce culture strength. Also wouldn't want to outright punish players for growing population in conquered cities and certainly don't want them to starve the population on purpose. (My guess as to the original intention behind ignoring the current population is that the designer saw it primarily from the perspective of a player trying to flip a border city – don't want culture strength to decrease as as food production tiles around the city flip.) However, eventually, when a city stays at a small population count, it's revolt chance should decrease accordingly, and using the highest-ever population directly after conquest contributes to the problem of a scarily high initial revolt probability. Hence the sawtooth curve.
	I agree that cities enveloped by foreign culture should flip quickly, but I don't want conquered cities to be untenable, and I don't want players to feel compelled to continue their wars in order to push back foreign borders. My rules are supposed to give the conqueror time to build up some culture.
	Slavery would be an easy way to get around anger from foreign culture, and to keep the population small, making the city easier to pacify. Also, Slavery could be used to raise units for revolt suppression in the very city that needs to be suppressed.
	Prior to AdvCiv 0.97, hurry anger (regardless of the number of angry citizens) had caused the population to be treated as 5 greater and there was no speed adjustment. (Note that anger duration is e.g. three times longer on Marathon than on Normal speed.) That turned out to be too punishing (and too implausible). Slavery should should be an option for urgent production orders, in particular Courthouses.
	That's all overly – and really unacceptably – complicated (already in BtS). It's OK for now; players don't really need to know the details. But eventually this should be greatly simplified.
See also	If 035 is enabled, it's *cultural* ownership (highest tile culture that matters for the revolt chance after 50 turns. 912d allows Slavery to be disabled. CFC post (response to Elkad) about the effect of hurry anger.
Tbd.	Maybe the penalty for using Slavery is now a bit too small. Losing one or even multiple citizens lowers the revolt chance substantially in small cities; the increased anger can't always (over-)compensate for that.
	Should population growth increase the city owner's city tile culture? A supressed population would typically not grow as fast as its supressors … However, this could encourage sacrificing or starving the city population.

Hover text for the nationality bar shows when sacrificed population increases foreign culture strength.

Rationale	Players who normally use Slavery all the time are unlikely to notice that revolt chance is lower without hurry anger; need some UI support for this. Ideally, players should be aware before sacrificing population, but I don't think they'd necessarily read hover text for the hurry button. They'll likely inspect the nationality bar though (sooner or later) when cities have much higher revolt probabilities than in BtS.
	I'm not showing *how much* hurry and conscript anger add to foreign culture strength. Culture strength isn't really part of the UI's vocabulary, so it would have to be expressed as additionally required garrison strength. In any case, it's awkward to implement and that time would be better spent re-designing the culture strength formula or replacing Slavery.

Barbarian cities flip on the second revolt. After a city is conquered or founded by Barbarians, it has has 0 revolt probability for 8 turns (adjusted to game speed through the Golden Age modifier). Barbarian units can suppress revolts just like non-Barbarian units.	Flip on the first revolt. No protection after conquest except that no revolt can occur under occupation (and cities can generally only revolt when within the culture range of the city's cultural owner). Barbarian units have 0 culture garrison strength (but this doesn't affect the occupation timer).

Rationale	Given the increased revolt probabilities in K-Mod and AdvCiv and the removal of the range restriction for revolts ([099c](099c)), a single Barbarian revolt happens (way) too quickly. A holy city conquered by Barbarians in the early game would flip back in something like 10 turns. Two revolts are still too quick, and, even for cities that can't flip, it looks strange if they go into revolt so easily.

I've tried some other remedies (letting Barbarians eliminate some of the former owner's tile culture upon conquest; increased chance for decreasing the occupation timer in Barbarian cities), but a grace period seemed like the most effective approach in tests since Barbarian conquests tend to happen in the early game when a Barbarian city can quickly generate some tile culture if they're not interrupted by revolts. |
| *See also* | [023](023) changes to the occupation timer. |

The revolt chance is increased on account of city religions if

a) the civ to whom the revolt culture belongs is alive, not a capitulated vassal of the city owner, has a state religion and that religion in present in the city; or

b) if the city owner has a state religion, and that religion is not present in the city but some other religion is.

In case b), the increase is only 2/3 of case a).

No change: The revolt chance is decreased (which may cancel out the above) if the city owner has a state religion, and that religion is present in the city. | Only a) increases the revolt chance, and also applies to capitulated vassals. Dead culture can't cause revolts. |
| In case a), the revolt chance is also decreased (but not by as much) if the city owner has no state religion.

AI city owner is slightly inclined to switch to the religion of the foreign population. | When there is a non-state religion in the city, then an oppressive state religion (which is not present in the city) is no worse than no state religion. |
| Hover text on the nationality bar shows when, on balance, religion modifiers increase foreign culture strength. | The UI shows only a revolt chance, no information about how it gets computed. |

Rationale	I almost removed this complicated stuff entirely, but it's kind of nice that spreading one's state religion can help flip a city. The BtS rule doesn't work for dead culture, so I felt that some replacement was needed to balance out the decreased revolt chance from owner's state religion. The BtS rule also doesn't work well for capitulated vassals of the city owner: The owner can just tell the vassal to switch to a different religion; not an interesting decision, and players that aren't aware of the religion/revolt rules get confused or screwed.

Reduced the religion-based modifiers (see above). Now at most +/-33% foreign culture strength	Religion can double or halve foreign culture strength.
Rationale	Spreading one's state religion had made a huge difference previously, going from doubled to halved foreign culture strength, i.e. a factor of 4.
Config	`STATE_RELIGION_MODIFIER` parameters in `GlobalDefines_advc`.

102	Show fewer foreign moves
Tbd.	Perhaps enforce "Quick Moves" for non-hostile foreign units.

AdvCiv	*BtS*
When a move starts in a visible plot and ends in an invisible plot, the camera is centered on the start plot and the player gets to see/ glimpse the unit moving out of sight. Not shown: Units that begin and end the turn in invisible tiles, i.e. move only through a visible tile.	Only moves ending in visible plots are shown. In the case of a multi-tile move ending in an invisible tile, not even the visible portion of the move is shown.
Rationale	Would make sense to show units passing through, but difficult to implement; can't catch the unit as it moves. What's easy to do is show the empty traversed tile, but that's confusing, and it doesn't make sense that the player learns about the tile but not the unit that moved. Better not to show those moves at all.
Tbd.	Often the unit moving out of sight is already gone when the camera jumps there. Not sure if this can be helped.
See also	Implemented based on 003k, which allows data members to be added to CvSelectionGroup.
AI avoids patrolling within its own borders and won't patrol beyond a 10-tile radius around its cities. Units for pillaging don't normally patrol. Patrolling units tend to move in a consistent direction. Leads to wider patterns.	AI constantly patrols within its own borders and those of other civs, especially with fast pillagers like Knights. Patrol movement is memoryless; narrow patters.
Rationale	The patrols make "show enemy/friendly moves" unusable, and there isn't much of a point. After all, borders grant visibility. I guess patrols add some uncertainty to small-scale surprise attacks (can't be sure how many units exactly are going to be near a city at a given time) and could help against nukes. Well, anti-nuke AI behavior needs to be improved at a later point anyway. Patrols in foreign borders make some AI visibility cheats less obvious, but that's not a good enough reason. Patrols in unowned land can keep Barbarians away from the AI's rivals; unowned land should only be patrolled if it's not too far away from the AI's cities.
Tbd.	I didn't disable inner-border patrols entirely because, in order to patrol unowned land, AI units may have to traverse owned land. Moving in one direction for a longer time also helps with that. That said, patrols in unowned land don't seem crucial either. Against Barbarians, the guard-city-site AI routine is probably more effective. The 10-tile distance check should perhaps also consider cities of (non-human) teammates and vassals. Might not be worth the extra computing time though.

Moves of non-hostile Workers, Missionaries and Executives within their owners' cultural borders are no longer shown when the "show friendly moves" option is enabled. As for non-hostile ships, moves of human ships and cargo ships are always shown (except AI ships on patrol), moves of other ships only when moving into sight, out of sight or inside the borders of the observing player.	All unit moves are shown.	
Rationale	These land moves are practically never interesting. Even if Worker stealing weren't disabled (change 010), I doubt that seeing rival Worker moves would help. For ships, it's a bit trickier because a passing Caravel could matter (target for Privateer, or could be carrying a Spy), but mustn't show patrolling Destroyers. Cargo units that can carry city attackers need to be shown in any case.	
	By not showing moves of cargo ships on patrol, I'm giving away some info about the intentions of the AI. That said, a lone cargo ship is practically always a patrol, and patrolling AI Galleys are just too annoying to watch.	
Config	`SHOW_FRIENDLY_WORKER_MOVES` and `SHOW_FRIENDLY_SEA_MOVES` in `GlobalDefines_advc`	
Tbd.	Could be an issue for enemy moves as well when there is city visibility from espionage. So, should perhaps apply the change to enemy moves too.	
~~Related: Fixed a bug that caused AI non-combat units to oscillate between safe cities when at war. See change 001.~~ (Also fixed in K-Mod 1.45, now using the K-Mod fix.)		
~~"show friendly moves" can be toggled using Shift + M. If the option is toggled this way, the change won't be visible in the Options menu, and won't be saved upon leaving the game.~~ (Disabled through a switch in `GlobalDefines_advc.xml`. Turned out I had never used this.)		
No moves are shown when in Globe view. If the Unit layer is active in Globe view, then the layer colors and indicator positions are updated as foreign units move (without a delay and regardless of the show-moves options).	The camera jumps around, but the moving units aren't visible in Globe view. If the Unit layer is enabled, the positions of the unit indicators are updated as foreign units move, but the layer colors aren't updated.	
See also	Requires 004m to be able to tell if the Unit layer is active.	

103	Spy unit can investigate cities		
AdvCiv	*BtS*	*Vanilla/Warlords*	
There is a passive mission that works as in BtS and an active mission with a base cost of 40 espionage points and guaranteed success. (For comparison, Counterespionage costs 100 and also guarantees success; as in BtS.) The AI doesn't use this new mission, i.e. still, to an extent, cheats with the information that is taken into account.	Can only investigate rival cities by accumulating espionage points (passive "Investigate" mission). That said, the "Sabotage Building" and "Sabotage Production" missions (and the cost of these missions) give away some information. The AI code for some of the active missions (e.g. "Foment Unrest") use information about the attacked city that the AI doesn't explicitly have, though that information can mostly	A local spy unit allows city investigation at no cost and regardless of whether it has moved. City religion can also grant visibility. No visibility from espionage points.	

Once the player closes the city screen, the Spy unit has its movement points spent but isn't teleported anywhere. The owner of the Spy can enter the city screen for free any number of times for the remainder of the turn. When investigating a city, the research bar, income and treasury are hidden. Slider settings are still visible.	be deduced. (E.g. the happiness level of a city is mostly a matter of civics and luxury resources, both public knowledge.) All successful active missions cause the Spy unit to be teleported to its owner's capital. When investigating a city, the owner's commerce slider positions, current research, gold in treasury and per-turn income are shown on the city screen.	As far as I remember, Spy units aren't teleported anywhere after a mission. Same info revealed as in BtS I suppose.

Config	Investigate mission added through `GameInfo\CIV4EspionageMissionInfo.xml`. Can remove it there or adjust its cost and success chance. To allow free investigation, one can set `bInvestigate` to 1 for the Spy unit and Great Spy unit in `Units\Civ4Unit Infos.xml`. I had done this until v0.95 along with a change in the DLL (which is still in place) that prevents spies from investigating if they've spent any movement points. After some testing, I felt that this made investigation too easy and added the active mission.
Rationale	To make Spies more useful for players that don't invest heavily in espionage. Also, constructing wonders is more fun when one can find out who else is in the race. Players don't spend thousands of espionage points just to find out if someone is currently building a wonder. In Vanilla/WL, investigation was too easy though, hence the cost. I've disabled the teleport so that players can investigate a city to find out its weaknesses and then execute one of the take-that missions on the next turn. No gratuitous info about foreign research because this would supersede the passive (and more or less reasonably priced) "See Research" mission.
Tbd.	Perhaps the cost needs to scale with the game progress somehow, e.g. the era of the city owner. Confusing: When inspecting a city that has just grown, the new citizen has no job at all (not listed as a Citizen specialist either).
See also	120d: Mission cost for Sabotage Production no longer shown on the Espionage screen. It had previously been possible to guess based on that cost whether a city was close to completing a wonder. 001: Bugfix in `CvPlayer::getUnitArtInfo` that shows unit art on the city screen in the owner's art style.

104	UWAI: See chapter Utility-Based War AI Most of the changes are in newly created classes. I've added a few auxiliary functions in pre-existing classes too. Technical documentation only inline. There are also some changes in unlikely places, which I've marked with the following sub-ids. One thing I guess I should explictly document somewhere in this manual: When an AI civ refuses to capitulate with the reason "Not right now", then that civ will capitulate on the next turn of the would-be master unless it can manage to improve the power ratio until then. This way, only the game state at the end of the capitulating civ's turn is decisive. If the capitulating civ thinks that the would-be master wants to end the

	war badly, then it may respond "Not right now" for several turns. Related [CFC post](#)
See also	[210a](#): UWAI changes that make the war trades alert less noisy.
Config	By checking "Aggressive AI (Legacy)" on the Custom Game screen, UWAI can be disabled. Can also disable it through XML, even in a running game (after saving and exiting), by setting either `UWAI_IN_BACKGROUND` or `DISABLE_UWAI` in `Assets\XML\ GlobalDefines_devel.xml`. UWAI settings in `AI_Variables_GlobalDefines.xml`.
Tbd.	Would be nice if the per-turn UWAI log file (configured in `GlobalDefines_devel.xml`) could be closed at the end of a turn, so that it can be deleted without having to exit the game first, but the logs are handled by `CvDLLUtilityIFaceBase`, which doesn't seem to allow files to be closed.
104b	Code for measuring path lengths for troop deployment from friendly cities to the cities of a (potential) war target. (A hack that measured distances between cities by generating paths for garrisoned units was removed in v0.94. Until v0.99, the `FAStar` pathfinder in the EXE had been used along with a tweak for aborting the search upon reaching a distance limit.)
See also	Now uses the TeamPathFinder described under [advc.pf](#). Some vague ideas by Nightinggale for faster pathfinding: [CFC post](#)
Tbd.	Until AdvCiv 0.99, scalability was a real issue with more than 18 players. That issue is now resolved insofar that AdvCiv (with UWAI) runs as fast (or at least nearly) as K-Mod even with 48 players. Nevertheless, the computation of deployment paths remains the biggest time sink on super-Huge maps. The pathfinding code can't be optimized much further. I've given Dijkstra's algorithm a try ([Git branch](#)) – which is potentially faster than A* when paths are needed for multiple destinations, but that turned out to be not much faster than Firaxis' FAStar class in the EXE and considerably slower than K-Mod's A* implementation in the DLL (which TeamPathFinder is based on). So, to save further time, fewer paths should be generated. The most promising approach is to consider only cities of the best potential war targets. There's a bullet toward the end of the [WIP](#) section about that. Two more ideas: Don't update distances for every one of our cities on every turn. Distances change gradually through tiles being revealed, routes constructed, border expansion and other factors – it should be OK to skip a city with, say, an 80% probability (adjusted to game speed). Major changes in diplomacy should result in an immediate full update: Open Borders, Vassal, Permanent Alliance, declaration of war. Only if we're directly involved in one of those. Also: upon the discovery of certain technologies. A tech is discovered every 4 turns or so, so doing it after *every* tech is undesirable. Try parallelizing the distance computations. On that note, AI war evaluations (based on the cached distances) could also be suitable for parallelization. (This one no longer sounds promising now that pathfinding has become much faster.) Take advantage of the triangle inequality: If we have computed a path distance from A to B and from B to C, we could use that as an approximation of the distance from A to C if B is a non-hostile city, i.e. not owned by the prospective war target or a current war enemy. One might first let a new class CityGraph compute a sparse graph with edges chosen based on area id, air distance and city size. Edges between our own cities will have to be allowed too. Graph updates: Local updates for changes that occur throughout a turn and a full update when updating the UWAI cache: Go through all relevant pairs of friendly and potentially hostile cities as before and make the graph compute a shortest distance using a standard graph algorithm (Kruskal/ Prim or something to compute all shortest paths). Turn that shortest graph distance into something slightly optimistic to account for direct paths being shorter than triangular ones.

104c	AI avoids military victory strategies in order to spare friends	
AdvCiv/ UWAI		*BBAI/ K-Mod*
When weighing its victory strategies, the AI considers whether a military victory (esp. conquest) will require an attack on a friend. If so, the AI will pursue different strategies instead.		The AI doesn't take friendships into account when choosing victory strategies.
If the AI still comes close to a military victory, it may ultimately attack friends, depending on the specific circumstances.		Once stage 4 out of 4 of a military strategy is reached, the AI ignores attitude when it comes to war targets.
Rationale	See the UWAI chapter. Victory strategies aren't within the scope of UWAI, but, in this case, the strategy weights need to be aligned with the DoW policy.	
Config	Only effective if UWAI is enabled.	
See also	[115](#): AI commitment to victory strategies	
104d	Changes to `AI_targetCityVal`. That function is not part of the UWAI component, but it's now based on heuristics that are part of UWAI (if it is enabled), and it feeds back into the UWAI projection of war outcomes.	
AdvCiv		*K-Mod*
Added a function `AI_assetVal` to CvPlayerAI that evaluates cities as economical assets, replacing code in `AI_targetCityVal` and `AI_cityTradeVal`.		`AI_cityTradeVal` (AI evaluation of cities for end-war trades) is almost just a stub in BtS, but has begun to overlap with `AI_targetCityVal` (AI evaluations of enemy cities to attack) in K-Mod. For example, K-Mod has added an evaluation of wonders to `AI_cityTradeVal` – which was already present in `AI_targetCityVal` in BtS.
See also	[advc.ctr](#) revises `AI_cityTradeVal`.	
Some changes to `AI_targetCityVal` that make an AI fighting or considering multiple wars focus more reliably on cities of rivals en route to a peaceful victory.		
Rationale	It's important that AI civs aiming at a military victory are able to pivot from war against a nearby target to war against a remote target that is getting close to a peaceful victory.	
Tbd.	Still no working very well I think. Part of the problem is that the militarily powerful civs often have very acrimonious diplomatic relations with their neighbors and pretty positive relations with remote rivals.	
The per-continent AI target cities get updated when the enemy status of a rival changes. This will also discard any target city of a vassal set by its human master ("we'll do our best").		K-Mod updates the best target city on each continent only every few turns, presumably in order to make AI offensives more focused.
Rationale	Looks like an oversight. It doesn't make sense to cling to a target city whose owner is no longer an enemy.	
Unless the military power ratio is very much in favor of the attacker, the AI prefers to target cities with low tile defense.		Tile defense factors into the targeting decisions of individual stacks, but is not a factor in `AI_targetCityVal` and thus not a factor for the per-continent target cities of an AI civ.
Rationale	While the per-continent target cities are only intended as intermediate targets, it still doesn't make sense to target e.g. a city with a Castle when the power ratio is nearly	

	even.
See also	cdtw.2 gives cities with very low defense extra weight as target cities when in the "Fast Movers" AI strategy.

Unless the military power ratio is very much in favor of the attacker, among potential target cities that require a naval assault, the AI prefers to target continents with few enemy cities.	In BtS/ K-Mod, `AI_targetCityVal` can't even tell whether a city will have to be assaulted from the sea.
Rationale	Should help the AI pick up lightly defended islands when an invasion of the enemy's main continent isn't feasible.

Make the AI somewhat inclined to target cities with strategic resources.	Resources near potential target cities are evaluated, but only from the point of view of the attacker.
Rationale	Very coarse – better than nothing. There is no (non-cheating) AI code for evaluating a resource from the point of view of another civ, and writing such code (and making it run fast) would be too much work. Also difficult to predict whether conquering a city will actually change ownership of nearby resource tiles; so I'm not even trying.

AdvCiv	*BBAI*
Increased the impact of distance. Now probably about the same as in BtS.	BBAI had reduced the impact of distance so that (comment in the code) it "*scales sensibly with map size*".
Rationale	The BtS formula was flawed (mainly when it came to naval attacks I think), but I don't think the map size should matter. Even on the biggest maps, the AI shouldn't generally target cities that are one or two dozen moves into enemy territory. (AI attack stacks will opportunistically attack cities along the way, but it's still not a good idea to try and cut deeply into enemy territory.)

104e	Halved military power values of ships through the DLL.
	Reduced military power (not combat str.) of Maceman from 9 to 8; Samurai at 9 (no change), Berserker at 9 (was 10), Jaguar at 5 (was 6). Increased power of Infantry to 22 (was 20).
Rationale	UWAI handles naval and land power separately, so the power of ships relative to land units doesn't matter. But the K-Mod AI uses a single power rating, and, considering that ships are only useful in certain situations, ships contribute too much power. Also, even if UWAI is enabled, some strategic AI choices are based on a single power rating, and the Statistics screen shows only one rating as well.
	High power of Maceman makes UWAI overestimate Maceman when predicting military build-up; appears as a better offensive unit than Grenadier. The bonus against melee isn't that useful for this type of unit at that point of the game (whereas 7 power for Crossbowman seems OK). UWAI (if enabled) increases the power values of all offensive units that can receive city raider promotions; this gives Maceman another boost.
	The 25% bonus of Infantry vs. Gunpowder is very valuable, whereas Woodsman I shouldn't be enough to bring Jaguar to the same power as Swordsman (which has 1 more strength). Montezuma is suicidal enough without overestimating his unique unit.
Config	`Civ4UnitInfos.xml`
See also	131 also makes some changes to XML AI settings, e.g. the asset value of Infantry. The BASE mod also halves the military power of ships (but I didn't get the idea from there). 131c changes some power values of buildings.

104f	More narrow conditions for Dagger strategy

No Dagger while in a chosen war; Dagger disabled entirely when UWAI is enabled.	Having started a war doesn't affect Dagger. The Dagger strategy can overrule the "Get Better Units" strategy; that's why the K-Mod AI can make Archer rushes.
Rationale	Staying in the Dagger strategy while at war doesn't make much of a difference for the K-Mod AI, but it gets in the way of my UWAI testing (UWAI running in the background while K-Mod decides).
	The purpose of the Dagger strategy seems to have shifted from BAI to BBAI to K-Mod. Here's the original description by Blake. The latest purpose was probably to build up units without having a war plan, and thus also without giving away war preparations through WHEOOHRN; then a sudden strike. UWAI solves this problem differently.
104g	Don't demand tribute from unreachable civs (implemented based on UWAI code)
104h	Moved the BtS and K-Mod code for negotiation of peace terms into a separate function, and made some minor functional changes (that also take effect when UWAI is disabled):
	If the winning side doesn't have much interest in continuing the war, gold (and no tech) is considered as reparations.
	The choice of tech for reparations is less randomized and more based on matching the target value for the reparations.
104i	Refusal to talk (RTT) when at war. A new AI memory type, `MEMORY_DECLARED_WAR_RECENT` is used for distinguishing a DoW at the request of a third party or through a defensive pact (DP) or a vote from a normal DoW.
Config	The `MemoryDecayRand` value of the new memory type determines the RTT duration in wars on behalf of a third party. This value can be set through `Civ4LeaderHeadInfos.xml`, and a default of 11 is hardcoded in the DLL (`CvLeaderHeadInfo::getMemoryDecayRand`). That default value corresponds to an expected RTT duration of 11 turns. (Though, as in BtS, war success and AI personality can shorten the duration.)
Tbd.	When the AI is already willing to talk according to the BtS rules, adding `DECLARED_WAR_RECENT` memory has no effect. Thus, a civ could e.g. make peace with the target of a force-war resolution right after the vote. Not sure what to do about this. For a start, the turn on which a vote takes place or a war ally is brought in should be remembered.

When a DP leads to a DoW, then declared-war-recent memory is added for all involved parties, not just those that join the war through the DP. When peace is made between two civs, the civs become willing to talk to the other side's DP allies.

Rationale	Otherwise, the target of the original attack would be free to negotiate peace, while the allies that declared war because of the DP would be forced to continue the war.
See also	kekm.3 allows DP despite war
Tbd.	War votes should arguably lead to a (10-turn?) period of forced war. The current code based on AI memory can't enforce war between two human civs (or just for 1 turn).

When two civs make peace, all their declared-war-recent memory is set to 0, i.e. both become willing to talk to everyone (unless war utility is too high).

Rationale	Refusal to talk is a matter of fairness toward a third party (sponsor, DP ally, voting members of the AP). When a war party bows out (possibly said third party itself), it's a big enough change in circumstances to disregard any commitments for continuing the

	war.
104j	Change in a master AI's war plans affects war plans of its vassals. The unit stacks decide where to invade first. Active even if UWAI is disabled, meaning that BBAI's `CvTeamAI::AI_isOkayVassalTarget` function ("*Block AI from declaring war on a distant vassal if it shares an area with the master*") no longer works.
Rationale	With K-Mod's AI changes, I doubt that `isOkayVassalTarget` is really needed, whether UWAI is enabled or not. I'm not allowing unit stacks to anticipate wars triggered by Defensive Pacts (DP) because diplo penalties need to be taken into account, so the strategic side of the AI needs to choose the target for the declaration of war and the tactical side needs to respect that decision.
See also	003b: It could be important for performance that `CvTeam::getMasterTeam` is precomputed.
104k	Moved rounding of trade values into an auxiliary function (no functional change)
104l	Caching of war utility to keep the user interface fully responsive when checking if a civ is willing to talk or willing to trade certain items. No functional change outside UWAI, but had to put some code to switch caching on and off into `CvDLLWidgetData.cpp`.
104m	Handling of tribute demands redirected to UWAI; fewer random AI requests. Regardless of whether UWAI is enabled: AI doesn't ask for help, an embargo or (no change) tribute if it's about to declare war on the player (war preparations complete). On Noble difficulty and higher, the AI only makes tribute demands with a significant trade value; e.g. won't just ask for a food resource or some small sum of gold. The AI can demand gold per turn (not possible in BtS). The demanded sum is based on the human player's commerce rate and expenses (not on the current gold income which is often negative or near 0).
Rationale	This should also address problems with players pillaging their own roads in order to cancel a deal that results from a tribute demand for a resource. That trick doesn't always work (the resource could be next to a river or in a city tile) and requires two roads to be pillaged, which is somewhat costly in the early game. So AI resource demands can make sense; don't want to disable them entirely.
See also	advc.ctr: The AI can also demand cities.

AdvCiv (regardless of whether UWAI is enabled)	*BtS*
The AI may demand multiple resources at once as tribute and resources that the human player has only one of aren't off-limits.	The AI demands only a single surplus resource.
The AI may demand the human player's map in addition to a technology.	Only one tech *or* the map.
Rationale A map or a single surplus resource isn't nearly valuable enough for +1 relations and a peace treaty.	
When the AI asks for help or tribute, it includes a peace treaty in the proposed deal.	The peace treaty gets signed as a separate deal in response to a "diplo event" that triggers when a human agrees to give help or tribute.
Rationale So that annual tribute and help deals can be identified (through the peace treaty) and canceled automatically. Also more transparent this way: The peace treaty appears on the trade screen.	
Config The display string for a peace treaty offered by the AI at peacetime is set by	

	`TXT_KEY_TRADE_SIGN_PEACE_TREATY` in `Civ4GameText_advc.xml`. Currently says "Sign Peace Treaty (10 Turns)" whereas, in other contexts, it says just "Peace Treaty (10 Turns)". Players have gotten confused (CFC posts 1 2) by the AI "offering peace" despite not being at war. Hopefully the word "sign" puts more emphasis on the offer being a (lasting) treaty. Don't want to invent a new term – it is very much a Peace Treaty like the one signed upon ending a war.
See also	City trades (advc.ctr) and war trades (146) also imply a peace treaty and show it on the trade table.
Tbd.	Should perhaps also add a peace treaty to the trade table as soon as a human player lets the AI side put something on the trade table if nothing has been added to the human side yet. When something else is added to the human side, the peace treaty should be taken away. So that a peace treaty is on the table if a human player asks for tribute or a gift. Or maybe this is too distracting ...
Peace treaties are canceled by `CvDeal::verify` at the start of a game turn – players never have to do it manually.	AI civs cancel experied peace treaties at the start of their turn, which usually means that the peace treaty remains in place during the preceding human turn unless the human player cancels it manually.
Rationale	More convenient in general, and will auto-cancel expired annual tribute and help.
An AI civ won't ask a human civ for help unless the human asset score is at least ¾ of the AI civ's asset score.	It's enough for the human civ's asset score to be half as much as the AI civ's asset score.
Rationale	Feels unfair to be asked for a handout by an AI civ that is clearly doing better.
104n	Diplo votes
104o	Handling of sponsored wars and conditions for vassal agreements when UWAI enabled.
104p	Target size for invasion stacks based on difficulty and less random (even when UWAI is disabled). Always choose a target city while war is imminent. (BtS sets no target with one chance in three.) Smaller target size on landmasses without capitals where the enemy has at most 3 cities.
Rationale	Except on high difficulty settings, the AI often took too long to get invasions started after finishing preparations. The change for small landmasses is really unrelated to UWAI; warfare tends to play out on a smaller scale there.
104q	Changed some K-Mod uses of `WarPlanStateCounter` to `AtWarCounter`
104r	Handling of empire split
104s	Propagate UWAI decisions about limited, total and naval war to other parts of the AI: • Treat faraway land targets as only reachable by sea. • Treat preparations for "total" war the same as ongoing or imminent "total" war in `AI_maxUnitCostPerMil`. • Change Area AI from Offensive or Massing to a naval assault type when UWAI thinks that the attack should be made via sea. • Increase train-invader chance when preparing a total war. (K-Mod: Chance only increased once the war has started.) • Tweak code that makes AI cities less interested in buildings and missionaries while preparing for war. War preparations now carry greater weight than an imminent or ongoing war. (But, overall, an ongoing war that goes badly still affects AI cities even more than preparations.) *Rationale*: Once stacks are coalescing or moving, cities don't need to focus as much on military production; but it tends to take the AI too

	long to get to that point.
104t	Handling of team changes: Permanent Alliances and elimination of a team member
104u	Fixes initialization issues with scenarios
104v	Messages about AI war preparations in all-AI games
See also	Chapter about all-AI games
104w	Make all the AI's cash and gold-per-turn (up to a limit based on total commerce) available for peace deals, and trust the UWAI code not to spend more than peace is worth.
See also	039 announces the conditions of peace deals to third parties. 036 and 550f make more gold available for trades other than peace deals.
104x	Loading of AI-related parameters from `AI_Variables_GlobalDefines.xml`

104y	No-war probability without Open Borders	
AdvCiv		*BtS*
If the no-war probability from `Civ4LeaderHeadInfos.xml` would be 100% for a civ that the AI doesn't have Open Borders with, the AI computes its no-war probability as if its attitude toward that civ was one level lower than it actually is.		If the no-war probability is 100%, then the AI tends to behave non-aggressively. Depending on AI personality, Pleased or Friendly attitude is needed for 100%.
Rationale	Don't want humans to take advantage of a friendly AI neighbor by closing the borders and thus, possibly, preventing the neighbor from attacking anyone.	

104z	Handling of random events that lead to war/ peace: "Spy Discovered", "Wedding Feud", "Great Mediator", "Heroic Gesture".	
AdvCiv		*BtS/K-Mod*
Spy Discovered – No change		There is some BtS code for deciding if the AI wants war, and K-Mod improves that code a bit.
Wedding Feud – Fixed a bug: When a response to a random event will anger a rival civ that the AI dislikes, the AI picks that response only if the rival is not too powerful.		The event data doesn't say that one of the responses can lead to a DoW; the AI only knows that the response is going to worsen relations.
		If a rival is powerful, the AI is all the more inclined to anger that rival. As a result the AI will, on rare occasions, get into a war with a far too powerful rival.
Great Mediator, Heroic Gesture – Reduced the probability for peace from 50% to 33%.		The DLL gets no info at all about this event, so the AI has to decide based on the "AIValues" of the two possible responses. Both have the same value, so the AI ends up picking a response at 50-50 odds.
No Change		The other side can still decide to continue the war but, if that's also an AI civ, it never does (no change in AdvCiv).

No Change	Through, what karadoc calls "an ugly hack" in the respective canTrigger functions, K-Mod prevents the event from triggering when either side refuses to talk.
Rationale	I don't want to invest time here because there are serious issues with almost all random events; the sane approach is really to disable them. The war/ peace events are especially consequential though, so I'm applying some more band-aids. I'd like to add that the war/peace events more or less hinge on irrational AI behavior. Giving the opponent an option to make peace is pointless – the two civs could just negotiate via the trade table. And Wedding Feud lets one civ pay 60 gold for a relations penalty, and the other civ is simply asked whether it wants to declare war (diplo penalties still apply). The proper way to handle the last three events would involve adding tags to the EventInfo schema so that the DLL can tell what the events actually do. (The DLL could probably also identify them based on EventTriggerData, but that's messy.) And `CvRandomEventInterface.py` would have to call some new Python-exposed DLL function instead of just always starting a war (in the case of Wedding Feud) or always making peace. If I wanted to overhaul the events, MongooseMod (see "Event Overhaul" in LunarMongoose's feature list) would have to be the starting point. This Rise of Mankind revision could also be of interest. I don't think it fixes the war/peace events though (or only through his DLL, which is closed-source). I'd tend to throw out the diplomatic events (which require the AI to role-play; better work more on the Dynamic Diplomacy changes instead). Similarly, tile yield changes could be better handled by separate mechanisms for depletion/ degradation, prospecting and propagation. Pandemics and uprising would also deserve separate treatment. In the end, mostly events that ask a player to pay gold should remain (though these will only make sense if binary research can be discouraged somehow) and events representing natural disasters.
See also	106g is responsible for recording war/peace events in replays. This open Git issue of the "We the People" mod mentions the problem of missing XML tags for certain random events. 137 tweaks the map-size scaling of quests.
~~"No Random Events" enabled by default.~~	All options on the Custom Game screen are disabled by default.
Rationale	I feel that at this point (v0.94), the mod makes improvements in so many areas that players might expect that Random Events have also been overhauled. So I want to signal somehow that this isn't the case. Renaming the option to "Random Events" (disabled by default) would be more consistent, but players who update from v0.93 would have to toggle the option then. As of v0.98, they're again enabled by default. I haven't improved anything about Random Events, but I've noticed that games started via "Play Now" always use the default options. My intention had only been to show the option as disabled the first time that a player enters the Custom Game screen.
See also	101 renames "City Flipping after Conquest" to "No City Flipping after Conquest". That was done earlier in the mod's development when updating was less of a concern.

105	AI relies less on `getAnyWarPlanCount`	
AdvCiv		*BtS*
Replaced most calls of `getAnyWarPlanCount` and some of `getAtWarCount` with a new function `AI_isFocusWar` that ignores wars that are probably not worth focusing on. Found values are unaffected by at-war status. (Other parts of the AI code already take care of overexpansion and Settlers not walking into war zones.) Settlers may move to join a group of units so long as that group isn't in immediate danger of being attacked.		The AI assigns lower utilities (found values) to settlement spots when at war, to the point that AI civs practically don't expand while at war.
Rationale	Just being at war doesn't imply that the AI needs to build up units or switch to wartime civics. It could be that the war enemy can't even reach the AI civ, or has just one city left, is hopelessly backwards etc. In particular, don't want human players to declare meaningless wars in order to hamper the economical development of an AI civ.	
See also	[001j](#) replaces `ALWAYS_PEACE` checks in AI code with `AI_isWarPossible` calls. [003b](#) has replaced the remaining calls to `getAnyWarPlanCount` and `getAtWarCount` with `CvTeamAI::AI_countWarPlans`, `AI_getNumWarPlans` and `AI_isAnyWarPlan`.	
Tbd.	Should've written an `int` function instead that puts the commitment to war efforts on a scale. That could e.g. also be used for the "serious war" check in `CvCityAI::AI_updateSpecialYieldMultiplier`.	
The AI makes no tribute demands when busy with a war. (Only relevant when UWAI disabled; with UWAI, such demands are possible.)		The AI checks if its own military power is greater than the target's defensive power before demanding tribute. The target's defensive power accounts for ongoing wars with third parties, but only in a crude, unreliable manner.

106	Announcements and replays
See also	Some additions to the replay log are covered by change [150](#) [004f](#) disables celebrations See [104z](#) about random events. Most messages about random events are irrelevant, but I don't want to fix this because I think players should just disable them. [127](#) records start and end of AI Auto Play in replays. [127b](#) stores jump-to info about most announcements (e.g. click a DoW announcement in the Event log to jump to the captial of the civ that declared war). [120f](#) announces and records spy missions that cause a revolution. [210](#): Civ4lerts [071](#) shows a message when first meeting a rival. [010](#) shows a message when a noncombat unit is killed through an attack. The "Dawn of Civilization" mod has added BUG settings for customization of announcements about Great People, religion spread and random events. [CFC post](#) (For AdvCiv, I think I have a sensible middleground and no need for customization.)

Tbd.	Would be nice to show and record messages about captured and razed wonders. I think Civ 3 had those. I've already written the code and translations (not on GitHub), but it turns out that `CvPlayer::acquireCity` isn't the proper place for my code because the raze-city popup isn't resolved until later. This really complicates things.
	The random events that spawn Barbarians should be announced even if the player that triggered the event hasn't met the active player. (Maybe, ideally, not if the active player's capital is on a different landmass.)

AdvCiv		*BtS*
Only GP births of known civs are reported. The city owner is stated if the city isn't revealed, e.g. "in the Incan Empire".		All GP births are reported; "in a faraway land" if the city is unrevealed.
Show only the GP name in yellow.		Entire message in yellow. Makes it harder to pick out the yellow BUG gold-trade alerts.

Rationale	Gotta kill some messages. This one is implausible (how do you learn about GPs on an undiscovered continent?), and doesn't have great strategic value.

AdvCiv		*BtS*
Finished wonders are always reported (no change); the city is stated if revealed.		Message says "faraway" when the city isn't revealed. If the owner is known, it is listed on the Statistics Screen. The message never states the city, but highlights it on the map.
No change to the way religions and corps are reported.		When a religion or corp. is founded, if the holy/HQ city is unrevealed but the city owner known, report only "faraway".
The name of the city where a GP is born is no longer recorded in replays. Record it as "The Great Scientist Antony van Leeuwenhoek has been born in the Russian Empire." (important info at the start and the end)		"Antony van Leeuwenhoek (Great Scientist) has been born in St. Petersburg (Catherine)!"

Rationale	"In the Incan Empire" is extra information. Reporting "faraway" (as in BtS) is misleading in this case. Not reporting them at all would also be bad. Should either report all GP births of a civ or none. The crucial info is how many GPs a rival civ produces, and not so much which GPs specifically.
	Wonders: just to be consistent.
	GP in replay: In order to shorten the message a little. The city is irrelevant.

Tbd.	Perhaps remove the GP name from replays, i.e. just "A Great Scientist has been born in the Russian Empire".

AdvCiv		*BtS*
Relocation of a civ's capital is announced to other players (but not recorded in replays). Regardless of whether the old capital was conquered or razed or whether a new Palace was constructed.		Conquered and razed cities are announced; without special mention of the capital. Constructed Palace isn't announced.

Rationale	Most of the time, it's easy enough to tell from the city name when a capital gets conquered, but shouldn't have to scroll around the map to find out the name and location of the new capital .
See also	[127]: In spectator mode, the announcement is shown regardless of map knowledge.

AdvCiv		*BUG/ Civ4lerts*

No more BUG alerts about civs no longer willing to trade a resource.	The alert about resource trades triggers whenever the set of resources offered by a rival changes from one turn to the next.
See also	[210e](#) makes the alert trigger also when the AI becomes willing to import a resource.
Rationale	It's a helpful alert, but fires too often. Typically, the willing-to-trade part is more important. The player then either trades for the resource or decides not to. In the latter case, it's not immediately helpful to know that it's no longer available.
Tbd.	The same reasoning can be applied to the tech trade alert, though techs seem to become unavailable less frequently. Unavailable techs and resources hint at inter-AI trades and/ or changes in AI attitude. The proper solution is to implement a new alert for AI attitude changes and one for inter-AI trades. (Update: The latter has been implemented by now, [210d](#), but turned out to be too noisy.) Will probably still need the "no longer willing" alert because the AI could also have started building a wonder.
Civics changes except to Free Religion excluded from the replay log. Replays record when a leader enters a new era. (disabled by default) Replays record the first discovery of each tech. Era transitions and first discoveries are shown in green. Cities founded are shown in the owner's player text color. Religion changes in light blue. Cities hit by nukes, meltdowns recorded. And report meltdowns to all players that know the city, and record when a city reaches Legendary culture.	Except for Liberalism, technological progress is only recorded indirectly through civics changes and constructed wonders. Cities founded in green, religion and civics changes and random events in white. Only announcements; not mentioned in replays. Meltdowns are only reported to the city owner.
Rationale	Civics changes are not very important and occur very frequently; make replays difficult to follow. (Religion changes tend to be rarer and more consequential.) Replays should give some indication of technological progress. Era transitions were my first idea, then I thought first-to-discover would be even better. There's some redundancy when techs cause religions to be founded, but I guess that can't be avoided. Colors: Green should be the color of research. For cities founded, I've tried magenta (the color used by the disabled city-founded Civ4lert) and `COLOR_CULTURE_RATE`, but they're too hard to read against the brown background of the Replay screen. I've also tried white (now that civics changes and random events are no longer reported), but found player colors easier to follow. (Although replay messages are otherwise not color-coded based on the affected civ; so it's inconsistent.)
See also	[ctr](#) uses `COLOR_CITY_BLUE` for city trade alerts.
Config	`SHOW_FIRST_TO_DISCOVER_IN_REPLAY`, `SHOW_ENTERED_ERA_IN_REPLAY` and `USE_PLAYER_TEXT_COLOR_IN_REPLAYS` in `GlobalDefines_advc`.
Tbd.	Perhaps recording every tech is too much and era transitions too little. Could do era transitions plus Liberalism plus first discovery of techs granting a free GP.
`EVENT_MESSAGE_TIME_LONG` is no longer used.	Extra long display duration for e.g. founded religions and corporations. Originally also for religion spread (duration reduced by K-Mod).

Rationale	Annoying; not a good way to draw attention (nor are the events in question that important).
See also	106c: Civ4lerts had also used `EVENT_MESSAGE_TIME_LONG`.

AdvCiv	*K-Mod*
Announce feature growth in the fog of war (so long as – no change – the nearest city belongs to the active player).	Feature growth is only announced on actively visible tiles. (It seems that BtS had announced feature growth even on unrevealed tiles – but you'd think that this would be a widely known bug. Maybe the announcement gets killed somehow when the tile is unrevealed.)

Rationale	Terrain features aren't subject to the fog of war.
106b	Message limit
Config	Since AdvCiv 1.0, this option is **disabled by default** because it's potentially confusing for players new to the mod. It can be enabled on the "General" tab of the BUG menu.

This change only applies when playing with the "Minimize Pop-Ups" (MPU) option.

Rationale	It's awkward when the Event Log opens along with a popup because then the Event Log can't be closed until the popup is dealt with. This happens all the time when playing without MPU. Also, I suppose players who don't use MPU are fast players that don't pay a lot of attention to each individual message.
Config	**Override:** `MESSAGE_LIMIT_WITHOUT_MPU` in `GlobalDefines_advc.xml`

AdvCiv	*BtS*
If more than 3 messages are waiting at the beginning of a turn, don't display them on the main interface; open the Event Log instead. Exception: Messages about most major events are always displayed on the main interface. The following major events are *not* shown on the main interface when the Event Log is opened: • Golden Age begun • city founded • city liberated, ceded • foreign city flipped • city captured/ razed • wonder built Also not shown, and now only a minor events: when a religion or corp spreads in a city, birth of a GP, civics changes, tech discovered. The signing of a Permanent Alliance is now a major event. Also open the Event Log when there are 1 or 2 messages along with a diplo popup (unless that popup is just an AI civ making first contact with the human civ). In Hotseat, always open the Event Log when there is a new message.	No limit on the number of messages. The Event Log is only ever shown when the player opens it. Major events are kept in the Event Log indefinitely, whereas minor events are cleared after 20 turns and info messages after 2 turns. The following events are major: War started/ ended, alliance formed, defensive pact signed, capitulation/ vassal agreement signed/ broken, tech discovered, first to discover Liberalism, circumnavigation, wonder/ team project built, city founded, civics or state religion changed, revolution started/ ended, Legendary culture reached, religion or corp founded, city religion/ corp spread or removed, city liberated/ flipped, game won, GW sustainability reached (K-Mod), vote source established (UN, AP), city captured/ razed, Golden Age begun/ ended, civ conquered, GP born/ GG killed, empire split, spaceship launched/ failed, nuke. The Event Log button becomes inaccessible when a diplo popup opens (hotkey also blocked). On-screen messages are cleared in between turns, including those that haven't been displayed yet. This means that a player who follows another human in the turn order misses most messages.

Rationale	The main interface really isn't suitable for displaying frequent messages. It's obtrusive; the display delays add up; messages can disappear too fast. My limit guarantees that after waiting for 3 messages, no further messages can arrive.
	When a diplo popup opens, any messages on the main interface become difficult to read, so it's important to provide access to the Event Log. The player should be able to learn what happened during the AI turns before making any decisions.
	Major events on the main interface: These can easily get buried and missed in the Event Log because they occur already during opponent's turns. Especially when playing without sound. The ones that I've excluded are either not that important or there is some additional mechanism that alerts the player (e.g. tech splash screen when a tech is discovered). Religions are spread all the time; that's not a major event at all.
Tbd.	Could repair message delivery in Hotseat; opening the Event Log is only a workaround solution. Would have to store messages within the DLL when they are triggered, and call `CvInterface::addMessage` only at the start of the recipient's turn. Can also rewrite that function in the DLL if necessary; probably just wraps data into a `CvTalkingHeadMessage` object and calls `showMessage`.
	Messages about major events don't appear on the main interface until all diplo popups are through. Same problem in BtS. `CvPlayer::postProcessMessages` already gets called before diplo popups are displayed and already tells the EXE to show major event messages. Still, when `CvGame::update` returns, the messages get held up by the diplo popups. This issue will hopefully be easier to resolve once AI-to-human diplomacy is moved to the start of the human turn (see *Tbd*. under 001e).
See also	The "We the People" developers are considering improvements to the message delivery system: GitHub issue. Doesn't sound like they'll add something that AdvCiv could adopt. Sorting messages by importance can mix up causes and effects.
	I got the following idea (and uploaded a mockup here): When there are more than three new messages at the start of a turn, don't to select any unit and don't display any popup; instead show all new messages in the unit command area and put all minimized popups in the info pane, along with a button to start unit cycling, instruction labels if needed, and any further buttons that could be helpful at the start of a turn (e.g. buttons for cycling through unhappy and unhealthy cities). Remove all those widgets when a unit is selected (but show them again when no unit is selected). Could also have little buttons next to each message for dismissing only that message, and tiny versions of the button graphic that comes with some of the messages (to communicate that the message can be clicked). Not sure if all this is doable in the SDK. Main problem: Diplo popups. I don't think it will be possible to give the focus to the command area or any panel therein when the Diplo screen is up, meaning that the player will be unable to scroll through the new messages before dealing with the Diplo screen, and there's only enough space for four messages at once. That's an advantage of the current solution: The Event Log can take the focus.

All messages are cleared from the main interface at end of turn.	Once displayed, messages only disappear when they've been on display for the proper amount of time (depends on message type).
Any messages still queued for display are flushed before that (i.e. cleared without having been on display).	When a human player's turn ends, all messages still waiting to be shown to that player are displayed at once (delay skipped).

Rationale	When turns are ended in quick succession, in BtS, messages from different turns get displayed together, which can be confusing. Dropping messages without ever displaying them isn't much of a problem anymore because the Event Log tends to open periodically and the missed messages are in there.

At the end of each human turn, a special message with text "Older messages:" is put into the Event Log. On the next turn, the player can tell from this message which messages are new. The special message never appears on the main interface and it expires after 1 turn, meaning that only one such message is ever in the Event Log.	Each message in the Event Log is preceded by a date. All messages with the current turn's date are certainly new, but those with the previous turn's date may or may not be new. E.g. tech discovered is sent at end of turn and appears at the start of the next turn.

Rationale	By the midgame, there are more than three messages on most turns, so scanning the log for new messages is something the player does all the time, and it's vexing when it's not clear which messages are new. Also easy for a message to slip by unnoticed. I've tried some things (backspace characters, setting the message turn to -1, same text color as the Event Log background) to get rid of the date in front of the special message, but it appears to be impossible from within the SDK.

106c	No more alerts upon loading savegame

AdvCiv	*BUG*
The BUGEventManager fires `BeginActivePlayerTurn` only right after the previous turn ends, not when loading a savegame.	`BeginActivePlayerTurn` fires after the end of the previous turn, and when loading a savegame.
Consequently, alerts aren't checked (and aren't displayed) when loading a savegame.	Alerts are checked after loading. Since Civ4lerts doesn't store data in savegames, the alerts assume a blank slate, and display a bunch of messages, e.g. for all technologies that anyone is willing to trade.
Alerts have the same display duration as normal events: 10 seconds.	Alerts are displayed for 20 seconds.

Rationale	Arguably a bug. `BeginActivePlayerTurn` shouldn't fire multiple times per turn, but that's what happens whenever a savegame is loaded. Listing all trades upon loading can't be considered a feature either. That info is presented better by the Foreign Advisor.
See also	[127](): Other changes to AI Auto Play
Tbd.	When cycling to another player with Alt+Z, BUG alerts fire. Should instead only do a silent check (and fire only after ending the turn). For the alerts added by AdvCiv ([210]()), this is already fixed; see comment in `CvPlayer::setIsHuman` – though the BUG alerts will require a change in Python. As a special case of the above, cycling after the creation of a colonial vassal can result in a (harmless) Python crash because Civ4lerts.py isn't yet aware of the new player.

106d	Civ4lerts and Scoreboard settings changed
See also	[004v]() also makes changes to the scoreboard. [kekm.30]() adds an option for civ and leader icons.

AdvCiv	*K-Mod*

Enabled some alerts by default, namely Worst Enemy and all those from the second batch ("MoreCiv4lerts") except domination and map trade.	K-Mod disables all alerts by default (whereas standalone BUG enables them all).
Enabled the "willing to talk" alert by default, and disabled "peace treaty" by default.	It's the other way around.
Refusal to talk not reported when it happens at the start of a war.	

Rationale	The ones I've enabled are essential for intermediate players, whereas domination and the first batch of alerts (city management and trade gold) are only essential for perfectionists.
	AI willingness to sign a peace treaty implies willingness to talk. Additionally, the "willing to talk" alert covers ending embargoes; very helpful to learn about those.

Changed the Advanced Leaderboard default layout so that open borders (B), espionage advantage (E), defensive pacts (D), trade network (N) and whether they have any vassals (V) aren't displayed.

Default line height lowered from 22 to 20.

Rationale	Frees up space on the main interface, and I find it difficult to imagine that players can't remember whom they have these relations with. The vassal indicator is redundant when "Group Vassals" is checked (as is the case by the K-Mod default).
Config	The in-game BUG menu contains a formatting string. See above for the letters I've removed (B, E, D, N, V).
See also	See 120h about the espionage icon. 085 shows B, D and N by default if "Expand on Hover" is enabled on the BUG menu (disabled by default).

Reduced the vertical space between Minimap buttons and Advanced Scoreboard.

Rationale	Wasted space. In K-Mod, with its smaller fonts but greater line height and the vertical space, the whole scoreboard doesn't even fit on my screen when playing with 18 civs.
Tbd.	When the BUG Advanced Scoreboard is disabled (default: enabled), then the space is now larger than before. I can't figure out where to change the text position in that case.
	The icons on the scoreboard are positioned a bit too high. The issue could be in BUG's `Scoreboard.py` or in `CvMainInterface.py` though I can't spot any obvious problem. If the tabular layout is disabled, the issue goes away.
	It would be very nice if additional icons were shown when mousing over a leader on the scoreboard – either just for that leader or for all of them. The format string should then support (through `BUG\Scoreboard.py`) a preceding underscore for info that is only supposed to appear while hovering.

AdvCiv	*K-Mod/Civ4lerts*
The "pending" alerts only trigger when a negative event is about to happen: when a city is about to become unhappy or unhealthy or is about to shrink.	Trigger also when a city is about to become happy or healthy or is about to grow.
	The non-"pending" city alerts trigger when a city becomes healthy, unhealthy, happy or unhappy or when a city shrinks or grows.

Rationale	When something bad is about to happen in a city, the player may want to do something to prevent it, and when something bad has happened, the player may want to do something to repair it. I think that's the purpose of these alerts. Presumably, the triggers for positive pending events were only added for consistency – most (all?) alerts trigger on some condition and the negation of that condition (e.g. willing to trade/ no longer willing to trade). However, reducing the noise from alerts is more important than consistency. I'm not removing any non-"pending" triggers because players may want to check the job assignment when a new citizen is born or when an angry citizen goes back to work. I don't think this reasoning can apply to has-become-healthy, but in this case, I don't want to break the pattern.
Config	Added a single BUG option for all three "pending" alerts that restores the K-Mod behavior. This has been requested in [here](#) (CFC).
See also	God knows what the "Borders Pending" alert was supposed to be good for. I've removed that to make space for [210](#).

AdvCiv	*K-Mod*
"Happiness Pending" (i.e. about to become unhappy) enabled by default.	All Civ4lerts are disabled by default in K-Mod. (I think BUG as a standalone mod enables them all by default.)

Rationale	"About to shrink" is often helpful to know too but annoying in recently conquered cities.
Tbd.	One reason why it pays to be alerted about cities that are about to shrink is that the AI (and thus the governor) doesn't take into account that cities can lose only 1 population per turn. Once it's decided that a city will shrink, the correct play is normally to deprioritize food for one turn. This is tedious, so the governor should do it automatically or there should be some additional penalty for a food deficit when the food store is empty.

AdvCiv	*BUG*
With the exception of the revolt alert, the city alerts can't trigger for cities under occupation.	I've seen "has become happy" trigger under occupation.

Rationale	Potentially confusing, and the new owner is probably going to take a look at the city once occupation ends, so there's no need for tracking its status in betweentimes.

106e	Report all religion and corp changes
Show a message whenever a corp spreads in a revealed city. Announce religion spread only if the religion differs from the city owner's state religion, or if the player owns the city or the holy city. Once the player is in the Industrial era, announce religion spread only if the player owns the city or the holy city.	Message only shown to city owner and to players that have the religion as their state religion or own the holy city (in the case of religion spread) or to players that own the HQ (corp spread).
When a corp spreads in a city that isn't revealed to the HQ owner, the city becomes revealed to the HQ owner.	HQ owner may receive messages about unrevealed cities.
Don't announce the establishment of a corporate HQ, and don't announce that the HQ spreads the corporation to the HQ city. I.e. show only one message when an HQ is founded, namely (e.g.) "Alexander has completed Mining Inc. in Athens".	Three messages: one for completing a wonder, one for establishing the HQ and one for spreading the corporation. (K-Mod already got rid of a redundant message upon expansion of a croporation.)

Rationale	Update (v0.91): The foreign religion spread messages get too annoying in the mid-/endgame; therefore disabled post-Renaissance. Perhaps there should be a "Show foreign religion spread" Civ4lert, but that would have to be enabled by default, and then players would have to manually disable it in the midgame – if they'd even realize that this is possible.
	Update (v0.92): It's actually also too annoying before Renaissance. No longer announcing if the religion is already the city owner's state religion.
Tbd.	Does a holy city generate gold from unrevealed cities?
Config	The era threshold for foreign religion spread messages can be changed through `STOP_RELIGION_SPREAD_ANNOUNCE_ERA` in `GlobalDefines_advc.xml`.
	The v0.92 update can be disabled by toggling `ANNOUNCE_STATE_RELIGION_SPREAD` in `GlobalDefines_advc.xml`.

106f	Announce canceled Defensive Pact (DP)	
AdvCiv		*BtS*
Canceled DP are publicly announced and recorded in the replay log.		DP are announced and recorded when signed; only the parties involved are notified of cancelation.

106g	No random events in replays	
Random events aren't recorded in replays. The replay says "in response to a random event" when a declaration of war or a peace treaty was triggered by a random event.		All random events are recorded in replays.
Rationale	Most of the random events are unimportant, and many are quite wordy, making replays difficult to follow. The war and peace events are exceptions.	
Tbd.	There are some other important events, e.g. fulfilled quests.	
See also	[104z](#) improves the AI response to war and peace events a little.	

106h	Replay lists game settings	
Most info from the Settings tab (Victory screen) is stored in the first entry of a replay.		The relevant info is stored in the replay file but not displayed anywhere. The Hall of Fame screen only shows difficulty, world size, starting era and game speed.
Rationale	For old replays and replays shared with other players.	
Config	`SETTINGS_IN_REPLAYS` in `GlobalDefines_advc.xml`	
See also	[004](#) adds custom map options to the Settings tab. Those options are included in the replay entry.	
Tbd.	Would be nice to indicate which color is the player color, but I don't think replays support the `<color>` tag, and I don't want to show an entire entry in the player color.	

106i	Replay file format
See also	[CFC post](#) by me about the format

No mod name is stored in replay files, meaning that AdvCiv replays can be viewed without loading (or even installing) the AdvCiv mod. By storing a mod id number in a part of the replay file that BtS ignores and through a hack (as `CvDLLUtilityIFaceBase::loadReplays` is not in the DLL), AdvCiv is able to load replay files without a mod name and to distinguish AdvCiv replays from unmodded replays.	The result of `gDLL->getModName()` is stored in replay files. When no mod is loaded, that call returns an empty string. All mods share a single `Replays` folder. The Hall of Fame screen loads all replays from that folder, compares the mod name stored in each replay file with the current result of `gDLL->getModName()` and displays only those that match. Double-clicking a replay file appears to bypass the name check, so there is a way to view replays created by a mod without installing the mod. But this doesn't work on Steam installations (not sure about Windows 10), and it doesn't work via the Hall-of-Fame screen.

Config	Can be disabled through `GlobalDefines_advc.xml`. Automatically disabled if a mod-mod adds map sizes, victory conditions, handicap or game speed settings. May also be automatically disabled if a mod-mod adds colors that BtS can't display; see comments in `GlobalDefines_advc.xml`.
Rationale	Mainly makes it easier to share AdvCiv replays with other players. Whether it's generally desirable to show replays from different mods on the same HoF screen is debatable. I think I'd prefer it, provided that replays from different mods are easy to distinguish, but I'd rather not inflict this on others. However, if the option was disabled by default, players would probably only find out about it when it's too late, i.e. when they have a replay that they'd like to share and that doesn't have the right format. (If a savegame from just before victory is kept, one can recreate the replay after enabling the option, but that's a long shot.) Player who don't like AdvCiv replays showing up in the BtS HoF can remove the offending replay files manually; the file names start with "[AdvCiv]" (see below).
See also	[003k](#) is a prerequisite as, otherwise, no data can be added to the CvReplayInfo class. 106 (see above) uses player text colors in city-founded replay messages. This doesn't make the compatibility issues worse because player colors are used in any case for the minimap and score graph. 106h (see above) ensures that AdvCiv replays are identifiable as such by showing the mod name in the first replay message. [106m](#) stores the minimap resolution in replay files.

AdvCiv is able to load unmodded replay files and those from other mods.

Config	Disabled by default; can be enabled through `GlobalDefines_advc.xml`. Not all replays can be loaded; see the comments there.
Rationale	Certainly nice to have the ability to show replays from any mod without having to install it, but this doesn't have to be enabled all the time (see the previous "Rationales" box).
Tbd.	If loading of non-AdvCiv replays is enabled, the HoF screen should show mod names somewhere and offer a filter-by-mod menu. The HoF screen is in the SDK, so this is doable – but a bit of work.

I've added some sanity checks to the function that reads the serialized replay data. When a game is started, loaded or exited, any replay data loaded by the Hall of Fame screen is cleared.	BtS relies entirely on a try-catch block to filter out files that don't have the proper format. After entering the Hall of Fame screen (from the main menu or at the end of a game), all replay data is kept in memory until the program is exited.
Rationale	Memory is generally a non-issue for AdvCiv, but there could be a lot of replays (especially if those from other mods are loaded too), and each could contain thousands of messages and minimap color changes.
Replay file names start with "[AdvCiv]".	The replay name is composed of the player name, year of the final turn and, apparently, a two-digit random number to avoid name collisions with earlier replays. BtS associates the `.CivBeyondSwordReplay` file ending with `Civ4BeyondSword.exe`, but this does not cause the proper mod to be loaded.
Config	`TXT_KEY_REPLAY_PREFIX_ADVC` in `Civ4GameText_advc.xml`
See also	[190a](#) shows the mod name on the Settings tab (Victory screen).
Tbd.	Known issues with the Hall of Fame screen: Crash to desktop when clicking directly next to a replay button (e.g. in between two buttons): `Microsoft C++ exception: __no_rtti_object` in `KernelBase.dll`; `msvcr71.dll` on the call stack. I get this crash also with unmodified BtS. It seems to occur before any of the `handleInput` functions in the SDK are called, so, if it can be fixed, it would have to be through the code that sets up the table with the replay buttons in `CvHallOfFameScreen.py`. Another Vanilla/BtS bug: Any replays listed so far down the table that scrolling is necessary can't be started. `CvHallOfFameScreen.py` doesn't receive any input when the replay button is clicked. It looks like the `addTableControlGFC` function (in the EXE) can't handle buttons outside the initially visible area of a table. The bug could then also affect the Domestic Advisor screen. There should be some way to work around this ... Temporary solution: Hover text added to the replay button that warns about the bug for entries beyond the 23rd. On higher resolutions (I have a height of only 1024 pixels), 23 may not be the correct value, i.e. the warning can be a false positive. K-Mod has fixed a somewhat similar crash; but I doesn't seem helpful here. [commit](#)
The column with the "start replay" buttons is hidden when viewing the HoF screen at the end of a game.	The column is shown but the buttons have no effect; can't start replays at that point.
106j	A deal canceled during an AI turn is only immediately announced if it's an open borders agreement, defensive pact or peace treaty; canceled gold and resource deals aren't announced until the start of the next human turn. Exception: If a resource deal is canceled because the trade connection was lost, the announcement is shown immediately.
Rationale	Only important events should get an immediate announcement. Canceled resource deals are (typically) followed by either a diplo popup or a declaration of war – unless the trade network was severed.

106k	City name changes in replays
At the end of each turn, the game checks if the name of any city has changed since the start of the turn. If so, the name change is recorded in the replay log.	City name changes are not recorded in replays. If a city is renamed, future replay messages refer to the city by its new name, but older messages remain unchanged.
Rationale	I'm recording the change at end of turn because a player who renames a city might change his/her mind once he/she sees the new name on the main interface; don't want to spam the replay log with temporary changes.

106l	Messages upon saving the game
Config	`AUTO_SAVING_MESSAGE_TIME` in `Assets\XML\GlobalDefines_advc.xml`
AdvCiv	*BtS*
No message when autosaving	Autosaving message at end of human turn
Rationale	Players who rely on autosave tend to set narrow intervals via `CivilizationIV.ini`, which makes the frequent messages disruptive. Autosave is known to be reliable; no need to report success. It would be nice to show a message when the interval is, say, greater than 2, but the DLL doesn't have access to that setting (and I don't want to write code for loacting and parsing the INI file).
Tbd.	Should add an option on the BUG menu when I merge the other autosave options from BULL.
"Saving ..." messages about quicksaves and regular saves are displayed for 4 seconds.	For 10 seconds, like most of the on-screen messages.
Rationale	Too long for such a short message that the player knows is about to appear. The message arguably also has the purpose of warning the player against terminating the process while the savegame is being written, but that can't take more than a couple of seconds.

106m	Replay screen takes up the full (display) screen
AdvCiv	*BtS*
The dimensions of the replay screen are set based on the display resolution.	Hardcoded to 1024x768 like most Advisor screens.
Rationale	For some of the Advisor screens, it can be helpful to see parts of the main map and HUD in the background and some can't easily be scaled up. Otherwise, as is in the case of the replay screen, there's no reason not to adjust them to the available space (except that it takes some work).
Credits	Somehow I never thought of scaling up the replay screen (easy to forget that it's in the SDK because neither the expansions nor BUG modify it) until I read this post by Kjotleik. I didn't ask him for his code because I'm assuming that it, again, only works for one screen resolution.

The idea to increase the minimap resolution came from CFC user VDNKh (link). |

When the `HOF_STORE_REPLAYS_AS_BTS` option (see 106i) is not enabled, then replay files store the `MINIMAP_RENDER_SIZE` value that was used to generate the terrain texture. This allows players to change the texture resolution without breaking compatibility.	The background texture for the replay minimap is (apparently) stored as bitmap with 8-bit color depth, an aspect ratio of 2:1 and a width determined by `MINIMAP_RENDER_SIZE`, 512 pixels by default. (I suppose the margins are set to black when the map doesn't have a 2:1 aspect ratio.) The game can only read replay files whose texture dimensions match the current `MINIMAP_RENDER_SIZE`.
Config	The XML settings mentioned above are in `GlobalDefines_advc.xml`.
Rationale	I haven't changed the default MINIMAP_RENDER_SIZE because I want to stay compatible with BtS. But at least players who don't care about that can use a higher resolution and maintain replay compatibility within AdvCiv. (Another concern with a higher resolution is that the same resolution is used for the minimap on the main interface, where the BtS resolution is sufficient and a higher resolution might carry a slight performance penalty. Switching back and forth between resolutions as the replay screen is entered and exited doesn't seem to be possible.)
Credits	VDNKh made me aware that maximizing the replay screen has made the terrain texture a bit blurry. (CFC post; under "requests")
See also	106i deals with other changes to the replay file format.
106n	Replays can show terrain texture from before the end of the game
If `REPLAY_TEXTURE_ERA` is set to an era number, then the minimap texture for the replay is stored when a player first reaches that era. The texture is included in savegames and written into the replay file when the game ends. This does not break compatibility with the BtS replay format. The texture has a size of 128 KB (unless `MINIMAP_RENDER_SIZE` is increased; see 106m); the EXE might compress that a bit when appending the texture (bitmap) data to a savegame. By default, no texture data is stored in savegames and the texture generated at the end of the game is written into the replay file (as in BtS).	Replay files include a terrain texture generated at the end of the game. That texture may show roads, railroads and the effects of deforestation and global warming.
Config	`REPLAY_TEXTURE_ERA` in `GlobalDefines_advc.xml`
Rationale	I'd prefer to use a pre-industrial texture, i.e. `REPLAY_TEXTURE_ERA=4`, because railroads are very noticeable (and blurry) on the maximized replay screen and didn't exist for most of the time period covered by the replay. However, I think using the final map is also a logical choice and getting rid of the railroads and global warming effects isn't worth a substantial increase in savegame sizes.
See also	This post outlines some obstacles against showing terrain changes in replays.

106n	Replays can show terrain texture from before the end of the game

If `REPLAY_TEXTURE_ERA` is set to an era number, then the minimap texture for the replay is stored when a player first reaches that era. The texture is included in savegames and written into the replay file when the game ends. This does not break compatibility with the BtS replay format. The texture has a size of 128 KB (unless `MINIMAP_RENDER_SIZE` is increased; see 106m); the EXE might compress that a bit when appending the texture (bitmap) data to a savegame. By default, no texture data is stored in savegames and the texture generated at the end of the game is written into the replay file (as in BtS).	Replay files include a terrain texture generated at the end of the game. That texture may show roads, railroads and the effects of deforestation and global warming.
Config	`REPLAY_TEXTURE_ERA` in `GlobalDefines_advc.xml`	
Rationale	I'd prefer to use a pre-industrial texture, i.e. `REPLAY_TEXTURE_ERA=4`, because railroads are very noticeable (and blurry) on the maximized replay screen and didn't exist for most of the time period covered by the replay. However, I think using the final map is also a logical choice and getting rid of the railroads and global warming effects isn't worth a substantial increase in savegame sizes.	

106o	Announce war/ peace changes for vassals along with their masters	
See also	002l supresses the war horns sound for declarations of war triggered by defensive pacts.	
AdvCiv		*BtS*
Declarations of war and peace deals that involve vassals are announced in a single message.		A separate message for each pair of teams that change their war/ peace status. E.g. 9 messages when a team with two vassals declares war on another team with two vassals.
Rationale	Makes especially large all-AI games difficult to follow.	
AI civs are less willing than in BtS to go on the defensive when the number of nearby enemy units exceeds the number of cities owned by the AI civ.		
Credits	From More Naval AI (Tholal)	
All AI cities try to add a stationary city defender beyond the first around the Renaissance era – except on Deity and Immortal difficulty (around the Medieval era). Coastal cities add a third defender in Renaissance (essentially no change).		In the Classical era, regardless of difficulty level. And another in Renaissance in coastal cities.
Config	The DLL sets the era bounds based on the unit production modifiers in `Civ4HandicapInfos.xml`. (So it's not really configurable, but the era number also isn't as hardcoded as the description above makes it sound.)	
See also	CFC post (item2) arguing that the AI leaves its non-border cities too lightly defended in AdvCiv on Deity difficulty. When this was written (AdvCiv 0.99), the extra defender came in Renaissance, regardless of difficulty. (And the first item refers to a mundane bug that has by now long been fixed.)	

Rationale	An extra defender everywhere is a big production sink now that AI production discounts have been reduced (through change [251]), especially in the early game. That forum post probably describes an AI enemy in the Classical era; so I could address that situation (to an extent) by adding the extra defender already in the Classical era on Deity. However, I don't really think that this would make the AI more difficult to beat overall as it's not all that often possible to sneak up on the AI through the territory of a third civ that the AI isn't afraid of. It's also a pretty clever stratagem that I don't mind being rewarded a bit. (The reward isn't going to be that great because the conquered cities won't be connected to the player's core territory.)
Tbd.	Perhaps shouldn't add a coastal defender when no threatening civ is known.

Reduced the impact of the era number on the number of floating defenders.	The baseline was something like 6 floating defenders per city by the Industrial era – in addition to the stationary defenders.
Rationale	This was probably the worst part of the BtS formula for floating defenders. Often, a third of an AI civ's units were floating defenders in the late game and thus unavailable for offensive operations (except through the "Crush" strategy).
When preparing a land war, the number of floating defenders is increased only slightly.	No distinction between a land war in preparation and an ongoing land war.
Rationale	Once war is imminent (preparations complete), there is usually still enough time to train some extra defenders before a counterattack is (maybe) launched.

AdvCiv	*BBAI*
An AI civ following the "Turtle" strategy stops building defenders when their number exceeds a threshold. As a result, the Turtle strategy seems to get used only rarely and briefly.	Turtling AI civs keep building defenders until they can make peace or the power ratio shifts in their favor.
Rationale	A turtling civ can't recover economically from building Archers for 50 turns.

AdvCiv	*BtS*
AI doesn't try to guard non-city tiles when a war starts looking hopeless.	Nothing to prevent guarding of resources or Forts. Not sure if an urgent need for defenders elsewhere can cause the AI to stop guarding non-city tiles.
Rationale	If they're strategic resources, it could actually be crucial to keep them, but I don't think that's generally feasible when the AI gets beaten badly. It's normally more useful to guard the cities then.
Reduced impact of Space victory stage 3 on city defenders.	Extra defenders in capital and all cities building spaceship parts.
Rationale	Space victory is a lengthy process, and stage 3 begins with the completion of Apollo. Doesn't usually mean that the AI just needs to survive in order to win.
When the AI computes the (spatial) closeness between two civs, cities on different continents are considered to be less close to each other.	Cities on different continents are considered to be closer to each other than those on the same continent. Closeness mostly affects the distribution of AI defenders, the Alert strategy, raze decisions and the price of sponsored war.

Rationale	Arguably an oversight by S. Johnson (or whoever wrote the PlayerCloseness code). The intention was probably to use a wider search range for cities on other continents, not to treat them as being nearer.
See also	In some situations, the impact of a land connection is further increased: 022: Alert strategy 104o: WHEOOHRN when already fighting a war (currently disabled)
The AI counts units that temporarily guard a city as defenders in certain situations.	Only units with the appropriate (permanent) Unit AI type are counted.
Rationale	I'm not sure how often this comes up, but I think the change can't hurt.
Credits	Merged from "RoM: A New Dawn" (Afforess). SourceForge revision: link

108	Starting site normalization and re-assignment
See also	For many maps, 027 handles the selection of starting sites will then also decide how to reassign the sites and which sites need to be improved through extra resources or features during normalization. kekm.35 deals with starting site assignment as well. 129 makes starts with multiple Gold or Gem resources or numerous Flood Plains much less likely.
Config	Several parameters in `GlobalDefines_advc.xml`; can (largely) restore BtS normalization too. Selecting "Balanced" on the Custom Game screen will also have that effect.

AdvCiv	BtS
Transform most but not all bad terrain (Tundra, Desert, Snow), bad features (Jungle) and mountain peaks around starting sites. Ice is always cleared. In multiplayer games, all peaks are cleared, too, and almost all bad terrain and features.	Ensure that all land tiles around starting sites are decent tiles.

Rationale	Removing all bad tiles makes the starting region look unnatural. A few bad tiles give a start more character and provide more variety without hurting the player much. (Until the midgame, some tiles typically remain unworked anyway.) ("Normalization" seems like a bit of a misnomer for these terrain upgrades. It fits better for the extra resources placed in order to reach a target found-city value. Anyway, it's all referred to as "normalization" in the code.)

Bad terrain is never replaced under a precious resources and a bad feature on a precious resource is only removed in the inner ring around the starting location.	Desert Gold always gets upgraded to Plains Gold, and Jungle is cleared from Grassland Gems.
See also	The "precious" resource class is introduced by change 129.
Rationale	High-commerce resources near the capital tend to be too powerful. If they happen to have a drawback, then it's for the better and that drawback shouldn't be taken away.

Stricter limit on the number of extra Forests placed around a starting site deemed too weak. Also don't place (too many) extra Oases when there are already a lot of Flood Plains or Oases.	Places Forests almost everywhere unless the Forests increase the found-city value sufficiently, i.e. if the site is no longer too weak.

Rationale	Usually, no number of extra forests will close the gap in found-city values, so BtS essentially creates a dense forest around every weak starting site. Too obvious. Could argue that Forests should increase the found-city value more when normalizing, but I don't think AI civs actually get that much out of Forests near their capital.
See also	I might switch the order of normalization and starting site reassignment in the future; see *Tbd.* under 027. Therefore the found-city value of Forests during normalization shouldn't depend on human/ AI status

When placing additional resources, resource types that already exist in the city radius are probabilistically avoided, especially in tiles adjacent to the existing resource, and especially for resources that don't normally form groups.	Resource types already exiting around a city aren't taken into account.
Rationale	To make starting sites look more natural. Also, clusters of food resources or Stone/ Marble don't play well; aren't normally allowed to occur for good reason. Weird resource clusters will still appear occasionally; good for variety's sake.

All resource placement during normalization is randomized. Tiles in the inner ring are preferred, but only probabilistically.	The placement of food resources (`normalizeAddFood`) is, shockingly, not randomized and K-Mod hasn't fixed that either. Meaning that, if possible, food always gets added east of the starting site. Specifically, Corn gets added. Thus, the inner ring has absolute precedence for food; for "extra" resources (`normalizeAddExtra`), tiles in the city radius are tried in a uniformly randomized order, meaning that the outer ring has a chance of 60% (12/20).
Rationale	Prefer the inner ring so that players have more leeway to move the starting settler.
See also	129 randomizes the order in which resource types are considered.

Normalization can add at most 2 hills; up to a total of 3 (no change).	Normalization ensures at least 3 hills within the city radius.
Rationale	For more variety; having just 2 hills isn't a major handicap. That said, starting sites with 0 hills initially are probably very rare.

Food resources (except seafood) and hills placed during normalization have to be placed on the same landmass as the starting site. Other extra resources also aren't placed on other landmasses – if it can be avoided.	BtS doesn't check this; may place extras on an island in the city radius. PerfectMongoose does have such a check.
Rationale	At least food resources can be crucial for making a starting site playable.

When the decisions about placing extra resources and features during normalization isn't made by starting position iteration (027), then the aim is to make the worst starting site at least 75% as good as the best in terms of found-city value; 80% in multiplayer games (no change).	Tries to make the lowest found-city value at least 80% of the best.
Rationale	Should magnify the impact of difficulty (`StartingLocPercent`) a little.

Starting locations near bad terrain are more strongly discouraged than in BtS and Ocean counts as bad in this context (K-Mod: half bad). Food resources are more strongly encouraged than in K-Mod (not sure about BtS).	
Rationale	To further steer starting sites away from extreme latitudes. They're not historically plausible and normalization is especially noticeable there. Ocean can't be fixed through normalization at all. Extra food placed during normalization is not so conspicuous; that change is more about not leaving sites with a lot of food without a starting site and, perhaps, to avoid having too much food on the map overall.
The city cross around the starting site is revealed when the game starts.	No tiles are revealed other than those that the starting units can see.
Rationale	Makes it more attractive to move the starting Settler. Don't want players to settle blindly and hope for the best (or regen) when a starting site looks poor. When all city-cross tiles are revealed upfront, and hidden resources are unlikely (see below), there isn't much to hope for. Can accept the starting site for what it is or go find a better one.
Config	`START_SIGHT_RANGE` in `GlobalDefines_advc`
In singleplayer games, initially hidden resources are disregarded when choosing and normalizing starting sites. Initially hidden resources are thus no more likely to appear around starting sites than anywhere else. (Less likely, in a way, because Horse, Copper and Iron can't be occur on rivers.)	As far as I can tell, yields from all hidden resources are fully factored into the found-city value when choosing starting sites. Only their strategic value is discounted. A poor starting site can hint at a hidden resource in the surrounding city cross.
Rationale	A poor starting site hinting at hidden resources breaks immersion.
	It seems that BtS and K-Mod don't even take into account in which era a resource is going to be revealed, i.e. Coal would be valued about as highly as Copper. This needed to be fixed one way or another.
	The flip side is that AdvCiv can theoretically produce very strong starting sites because `addExtras` is unaware of hidden resources. That's something to keep an eye on. That said, at least for the AI, the city radius of the capital isn't much of a predictor of, say, the a civ's score in AD 1.
The AI moves its starting Settler if it sees a better site on turn 0 after moving its exploration units (if any). Most of the time, the AI still founds its first city on turn 0 (possibly after moving the Settler just 1 tile), and it practically always has a city at the end of turn 1.	The AI always founds its first city on turn 0 without moving the Settler.
I've made a few adjustments in the found-value function to help the AI evaluate potential sites for its capital. In particular, a low-ish value is assumed for unexplored tiles. This change does not apply to scenarios; AI always settles in place as in BtS.	There is special code in `AI_foundValue` for selecting starting locations, but that's not really AI code. E.g. it evaluates also tiles beyond the city radius. Unexplored tiles are normally treated as having 0 value.
Rationale	The starting site isn't assured to be locally optimal, and moving the Settler is sometimes, rarely, very advantageous. (More often, moving the Settler is only slightly better than settling in place.) Put differently: The AI's ability to move the Settler means that the starting site selection and normalization code doesn't have to try hard to find or create a local optimum.

Credits	Merged from the Better BUG AI mod, mostly by copying the code by Fuyu and Afforess. I've made a few adaptations and added some special clauses to the found-city value function because neither the tile evaluation for starting tiles nor the normal evaluation was quite right for this AI decision.	
AdvCiv		*Better BUG AI*
The AI is reluctant to move its starting settler farther than 1 tile on crowded maps.		It can happen that can AI civ moves its settler very close to another civ's capital. Since borders haven't expanded yet, the AI won't necessarily be aware of the nearby rival capital.
Rationale	Capitals at a close distance can be a major (random) handicap for both civs involved.	
Credits	Elkad reported a game that (s)he quit after AI Willem had founded his capital at a very close distance. link	
AdvCiv		*BtS*
Exploration units that civs receive at game start are placed on a different tile than the starting Settlers, Workers and defensive units. If more than one exploration unit is received (AI on Immortal and Deity), each one is placed separately.

Since the starting Warrior that humans receive counts as an exploration unit, nothing changes for human civs under typical game settings. If a human civ receives free defensive units or Workers (low difficulty settings or later-era start), then there is a slight change: Those units are placed on separate tiles in BtS; now they're placed together with the Settler(s).

Exploration units can be placed at most 3 moves away from the starting site. | | All units that an AI civ receives at game start are placed in the same tile. For humans, only the Settlers are placed in the same tile; every non-Settler unit is placed alone on a separate tile. The UnitAI type (City Defense, Explore, Worker) of the non-Settlers doesn't matter.

Can't be placed on a different landmass than the starting site, but can, in rare circumstances, be placed across a large bay, i.e. a long (walking) distance away from the future capital. |
| *Rationale* | My guess is that the separate starting tiles for human units were implemented in order to nudge beginning players away from immediately fortifying the starting Warrior in the first city. The separate placement also reveals a few more tiles and thus makes it easier to move the starting Settler; I like that (though it may not have been the original goal) and, now that the AI knows how to move its starting Settler, I want the additional visibility also for the AI.

And of course it's always nice to make the rules for human and AI civs more alike. | |
Config	BtS has a Python callback "startHumansOnSameTile" that can make human units start in a single tile. That should still work, but won't affect AI units.	
See also	The path distance check is based on pathfinding code written for 104b.	
Resources added during normalization can't result in more than 3 seafood; more than 2 unlikely. Coastal seafood resources now count as "high-food" resources.		More than 3 coastal food ruled out, and more than 3 ocean food ruled out, but both taken together practically unlimited (up to 6).
Rationale	Cities with more than 2 seafood grow unrealistically fast; doesn't play well either. This happens too often.	

Tbd.	When I get around to XML balance changes, I intend to reduce the food yield of Pig Pasture and Fishing Boats by one. Should make starts with overabundant food less common.
See also	[001](#): Whale doesn't count as a food source at all.

Try to avoid placing food as extra resources when a site already has enough food and just needs to be buffed.

Rationale	As above; don't want sites with very much food that often. Note that a high proportion of resources allowed for normalization are food resources.

Gold can't be placed during the normalization step.

Config	`bNormalize` flag in `Civ4BonusInfos.xml`.
Rationale	There's a pretty good chance that a Gold resource will turn a relatively weak site into one that is too powerful.
Tbd.	Don't want to block Gems entirely, but would probably better not to place them at rivers during normalization. (I wouldn't want to prohibit river Gems in general.)

AdvCiv		*K-Mod*
Reduced the chance of lakes as extra freshwater sources (in favor of rivers).		30%-50% chance of a lake instead of a river, depending on whether the starting tile is coastal. (And civs will also start at a lake when there happens to be one already before the "normalization" step and when no river can be placed. In BtS, starts next to a lake had occurred only in these circumstances.)
Rationale	Lakes do have the advantage of granting freshwater to up to 8 tiles while a coastal river (single segment) only affects 2 or 3 tiles. Therefore lakes make it easier for players to move their starting settler and support more Farms. I'm still dialing the lake chance down because most of the early civilizations emerged near rivers.	
When adding a lake for freshwater, tiles adjacent to the starting site are (strictly) preferred. BtS had also done that, but K-Mod had randomized the placement of lakes. No freshwater source gets added if there is already a freshwater tile adjacent to the starting site (i.e. a freshwater source two tiles away from the starting site).		K-Mod randomized the placement without regard for distance from the starting site. karadoc was aware of that: "*[S]ince the lake placement can be anywhere in the fat cross, it is less likely to be adjacent to the city and so that's why you might see fewer fresh-water starts.*" [source](#)
Config	If `NORMALIZE_STARTPLOTS_AGGRESSIVELY` is set in `GlobalDefines_advc.xml`, then the game tries to guarantee a freshwater source adjacent to every starting site.	

Rationale	The K-Mod change may have been partly motivated by convenience, but I think karadoc also wanted to increase the variety of starting locations. I'm in favor of that too, especially now that the AI knows how to move its starting settler. Though even if a player chooses not to move the settler onto a freshwater tile, adjacent freshwater can still be useful for irrigation.
	I didn't like that only lakes (not rivers) were being placed two tiles away and I think it happened too often with lakes (about a 12 in 20 chance when placing a lake). Also, two freshwater sources close to a starting site can look a bit strange. (Can still happen naturally, i.e. prior to normalization.)
Tbd.	Place a 2-tile lake in some situations?
108b	Starting site swaps based on difficulty
See also	027: If starting position iteration is used, then the volatility value of a starting site can also affect whether the site gets swapped and to whom.

AdvCiv		*BtS*
Slightly increased `StartingLocPercent` for most difficulty levels, meaning that human civs receive slightly worse starting locations than in BtS. E.g. on Noble, humans receive the median location if the total number of civs is odd; one better than the median if the total number of civs is even.		The starting locations selected by the map script are reassigned by the DLL based on the difficulty setting. `StartingLocPercent` is a percentile rank for the human civ. On Noble, humans always get locations that are slightly better than the median (40% rank), i.e. they get an advantage over the AI civs. (In as much as the game can't accurately determine whether one starting location is better than another.)
Rationale	Noble should only favor human or AI when there is an important reason to do so. And I like giving humans rather bad starts on the higher difficulty settings; it's a more organic handicap than e.g. free tech (see also chapter SPaH about this).	
See also	250d slightly reduces some AI advantages from the difficulty setting.	
Config	Through `Civ4HandicapInfos.xml`; though I've also changed the way these values are handled by the DLL a bit (see below).	
If the difficulty level sets `StartingLocPercent` to 100 (Deity: 95), then the human players receive the worst starting sites.		Human players only ever receive the second worst starting site.
Rationale	Could be a bug. Or the developers didn't trust their starting site heuristic and worried about players getting unplayable starts. This shouldn't be an issue with K-Mod (rewrites the starting site heuristic).	
	Doesn't really matter since `StartingLocPercent` isn't 100 on any difficulty level.	

Except in scenarios that assign specific starting locations to specific civ leaders, the starting sites assigned to human players are selected based on the difficulty setting. In team games, an initial assignment of starting sites is made either by SPI (see 027) or through the BtS algorithm (see on the right; with a couple of tweaks). Then, if all teams have the same size, starting sites are swapped (pairwise) between teams so that the difficulty setting takes effect without causing team starting sites to become dispersed.	In team games, a round-robin algorithm is used to assign to each team starting sites of a similar total value; afterwards, sites may be swapped around to bring members of the same team closer together geographically. In non-team singleplayer (no change), human starting sites are assigned based on difficulty (`StartingLocPercent`). In non-team multiplayer, humans receive the best starting sites.
Rationale	I see no reason not to use `StartingLocPercent` in multiplayer and team games. (Other than the implementation effort to make it work for team games.)
See also	135c: other multiplayer changes CFC post about issues with the BtS algorithm for assigning starting sites to teams.
If a map script calls `useDefaultImpl` in `assignStartingPlots`, the DLL reassigns those sites through swaps based on team membership and difficulty (see above).	The DLL will select new starting sites (based team membership and difficulty) and ignore the sites selected by the map script.
Rationale	This seemed like the easiest way to let PM use the standard reassignment algorithm.
See also	Chapter about PM
If a map script overrides `findStartingPlot` but not `assignStartingPlots`, then the DLL swaps starting sites around unless a custom map option is set to the value "Historical". (If `assign StartingPlots` is overridden, then the DLL doesn't swap anything; see previous blue box.)	Overriding either `findStartingPlot` or `assignStartingPlots` means that the DLL has no part in the assignment of starting sites.
Rationale	In principle, a map script that overrides `findStartingPlot` may choose specific starting sites for specific players, but none of the official and bundled map scripts do that – so, generally, it's fine to make swaps in the DLL. In my opinion, a script that assigns player-specific sites should override `assignStartingPlots`. However, the EarthEvolution3 map script – which is essentially a real-Earth scenario wrapped into a map script – only overrides `findStartingPlot`, and has an option for "Historical" starting sites (enabled by default) that breaks if the DLL swaps locations around. Hence the hardcoded exception.
See also	CFC post reporting EarthEvolution3 being incompatible with AdvCiv 0.99 (fixed in v1.0).

109	AI improvements for isolated starts	
AdvCiv		*K-Mod*

AI civs that haven't met any rival by the second era focus more on Science, and less on the flavor values set in `Civ4LeaderHeadInfos.xml` until they meet a rival.	Optics is prioritized when isolated but not Science in general.
Isolated AI civs are reluctant to pursue a culture victory. Adopt the "Economy Focus" strategy when alone, or when no threatening civ is known.	AI civs that haven't met anyone adopt Economy Focus; I don't think that strategy is reliably adopted when there are Friendly or faraway neighbors. Economy Focus (a BBAI-introduced strategy) reduces the production of military units.
Rationale	All naval technologies have a Science flavor, and faster research will reach Optics faster.
See also	130n bases religion hate on the number of known adherents. 130p decouples enemy trade penalties from the has-met counter. Both should make it easier to find partners abroad.

110	Changes to AI military build-up
See also	104s increases military build-up when preparing for a "total" war. 017: AI trains fewer military units in general
AdvCiv	*K-Mod*

Increased the target commerce surplus for research in the early eras of the game. (If the target isn't met, the AI is reluctant to build more units, including Settlers.)	The same target for all eras.
Gold reserves are taken into account (if the AI has a lot of gold).	Only a matter of the commerce rate and expenses.
Added a function `AI_financialTroubleMargin` that says how close the AI is to being in financial trouble. So far used only in one place: To make the city AI choose gold buildings more often.	`AI_isFinancialTrouble` is a boolean function.
Rationale	Don't want the AI to expand too rapidly in the early game and fall behind in research.
Credits	The code for the gold reserves is from the Caveman2Cosmos mod (by Koshling). SVN revision The change to the city AI is from "Rise of Mankind: A New Dawn" (by Afforess) via this SVN revision by alberts2.

AI production weight lowered to 220%. Was 270% of the commerce weight (except when in financial trouble).	
AI food weight lowered, in part, to match the reduced AI use of Slavery (change 121b).	
Rationale	Still high, but the AI is better at growing cities than at choosing sensible tech or city production. Would make sense to increase the production weight when Slavery is used less, but 270% was way too high, and resulted in post-Chemistry Workshop spam (without supporting civics). Huge outdated AI armies don't exactly make the game enjoyable either. In the late game, it doesn't normally make sense to grow cities further. I'm still giving food a fairly high weight because the AI should only avoid growth, not actually shrink cities (leave that to human players who know what they're doing).

Per-turn limit (equal to the current era number) for the number of military units that the AI may disband when in financial trouble. (Doesn't apply when already in strike.) When in financial trouble while losing a war, the AI may still disband units but is hesitant to do so.	When in financial trouble, the AI will immediately disband as many units as it takes to reach its desired upper bound on military spending. The AI won't disband any military units when losing a war (except when forced by the rules in a strike).

Rationale	Allowing the AI to disband an arbitrary number of units is clearly dangerous. Income may well increase (gradually) in response to financial trouble. Not disbanding at all in a bad war will eventually lead to strike (and then the AI doesn't get to pick the units).

So long as the AI has any undeclared war plan, it tries to cap its military spending.	

Rationale	Declaring war will often reduce the funds of the AI through canceled trades or angry citizens, and unit supply can add to expenses (though the AI gets a 50% discount on supply costs). Declaring and then having to disband units is really bad.
See also	vedg posted a savegame where this happens to the K-Mod AI.

When deciding which units to disband, the AI takes unit locations on the map into account (in a rudimentary way – plot danger, city threat, mission target plot).	Based on production cost, XP, unit AI type and mission AI type. And every city has to retain at least one defender.

Rationale	Try not to disband units that might be about to die in combat.

111	Changes to pillaging
See also	004: Pillage gold can't randomly be 0. 004c: Minor tweak to group pillage missions. 004g: Announce pillaged tiles immediately. 004k: Disable Sea Patrol mission 005c: Can't pillage City Ruins. 033: Always-Hostile units can't pillage vassal's, master's tiles. 064d: Block exploit involving pillaging of own resources.

AdvCiv	*K-Mod*
When pillaging in friendly territory, routes are pillaged before improvements.	K-Mod allows pillaging one's own route (BtS doesn't; but I think it allows improvements to be pillaged), but improvements always get pillaged before routes.

Rationale	Pillaging one's own improvements can, in theory, be useful as part of a scorched-earth tactic. I've never once used that though and never read about anyone using it. It seems that roads do get pillaged occasionally. The K-Mod changelog mentions Forest Preserves. Knocking out a road can also disconnect Coal without destroying the Mine (that said: Coal at a river can only be disconnected by destroying the Mine), or can slow down rival units. Or maybe the best argument is that it's logical to destroy the cheaper structure first.
Credits	Inspired by this post by CFC user P&enny.
See also	This CFC post suggests that players may not even realize that they can pillage their own roads when improvements take precedence.

AdvCiv	*BtS*
Can't pillage routes on unowned tiles that contain a (non-Spy) unit of a different team.	No special restrictions for pillaging unowned tiles. Can, in particular, pillage routes in order to slow

	down another civ's units.
Rationale	Not an important change. Yanking out roads from under other civs' stacks is pretty bizarre, and occasionally exploitable.

112	AI changes for voluntary vassal agreements (VVA ; "peace vassalage")
Tbd.	The VVA code was already messy in K-Mod, and it's worse now. Should be rewritten based on a utility value that replaces the dozens of exclusive clauses.
See also	037: Rule changes to prevent masters from sabotaging voluntary vassals.

AdvCiv	*K-Mod*
AI lowers its VVA attitude threshold (Friendly or Pleased for most leaders) only when feeling powerless (as in BtS) *and* acutely threatened, particularly when in a losing war against a third party.	AI lowers its attitude threshold when it is among those civs with the least military power.
A civ that has at least one Defensive Pact and is not at war with anyone refuses to sign a VVA.	Defensive Pacts don't matter for VVA decisions. Once a VVA is signed, Defensive Pacts of the vassal get canceled.
No leader is willing to sign a VVA when Cautious and not threatened. Leaders that have their threshold set to Cautious in XML are instead a bit more willing to lower their threshold when threatened.	E.g. Frederick signs a VVA at Cautious.

Rationale	VVA happen too quickly in K-Mod, probably also owing to increased military budgets. The idea that civs without prospects for winning the game should become vassals asap comes from Warlords though. Perhaps they're supposed to catch up under the protection of their master, and then break free again, but it doesn't usually work this way. The AI should certainly prefer independent survival over becoming a vassal and helping another civ win.
See also	133 cancels tribute deals once vassalage ends. 143 adds recently-canceled memory for vassal agreement.

AdvCiv	*BtS*
"Grown-too-powerful" restriction removed. On the contrary, when a civ gets close to Domination, its vassals only cancel the agreement if they're getting close to a peaceful victory – so long as those vassals are at least Pleased with the master.	AI leaders refuse to sign VVAs when the would-be master gets close to a Domination victory: "You've grown too powerful for us." I think they may also cancel the agreement when the master approaches Domination.

Rationale	Just prolongs the inevitable. (It does look strange when the last non-vassal player in a game capitulates, but, even then, it's better not to drag the game out.)
"Your land is too far away" rules out vassal agreements only until the Industrial era. If the prospective master is in the Industrial era or later, its power rating is decreased when not sharing a continent with the vassal, but a vassal agreement is possible.	The AI refuses to become the vassal of any civ that it doesn't share a continent with.

Master refuses to accept voluntary vassal whose population is much smaller than the master's unless aiming for Conquest victory or liking the vassal.	Only considers refusal when acceptance means war.
Rationale	Peace vassals are fickle and may lead to wars with third parties. Not worth the hassle unless they bring a considerable economic value.
When deciding whether a vassal breaks free, the territory of the vassal is treated as at least 10 tiles large. Thus, a vassal that has fewer than 15 tiles left when becoming a vassal can only break free by gaining land, not by losing land.	A capitulated vassal with, say, 12 tiles initially gets to break free after losing 6 tiles (from war or culture pressure). Voluntary vassals don't care if they lose tiles.
Rationale	Through change 143 (cancelation of voluntary vassal agreements), vassals with very little territory left can end up changing hands a lot.
Civs that are at stage 3 or 4 of any victory strategy refuse to sign vassal agreements, and break free if they can: "We'd rather win the game." The leader of AP or UN also refuses/ breaks free.	Civs that are close to domination victory refuse to become vassals (such civs will normally also be way too powerful to accept, so this is pretty pointless). In K-Mod, civs refuse to accept vassals that are at stage 3 or 4 of Culture or Space victory ("Surely, you must be joking"); may also cancel the agreement. Diplo victory not covered.
Rationale	Don't want a civ to win the game while hiding behind a master who can't cancel the agreement, especially not a human master. Important to let the vassal refuse (not the master) because the master could be human.
See also	115 prevents civs with too little production capacity from pursuing a Space victory, which is important for this change: Technologically advanced civs might otherwise refuse to ever capitulate once they have the Apollo Project. 143b prevents vassals from having nukes. 014 prevents capitulated vassals from pursuing victory strategies.
A colonial vassal can break free like a normal voluntary vassal, but the vassal's personality has less impact and stricter piower and attitude thresholds apply.	Only attitude can cause an independent colony to break away. And since the +10 "granted us independence" bonus doesn't decay, this happens practically never.
Rationale	A War of Independence should at least be a possibility.
See also	130r causes "granted us independence" memory to decay.
An AI civ ready to become a vassal contacts the prospective master only with a per-turn probability. The probability is based on the scoreboard rank of the master – between 1 in 20 if the master is ranked in the middle of the scoreboard, and 1 in 40 if the master is at the top. Increased by 400% if at war with anyone.	The same contact-delay is used for vassal agreements as for permanent alliances, but this only affects offers to human players. There is only a 1 in 80 chance of implementing a permanent alliance but no such probability for voluntary vassal agreements; they're checked each turn and are directly implemented.

Rationale	It's possible that the Warlords developers had meant to use the permanent alliance probability also for vassal agreements and misplaced a closing curly brace. Be that as it may, civs are too quick to sign vassal agreements when their power ratio takes a dip, which doesn't only happen in defensive wars, but also when an AI focuses on its economy (AI strategies Economy Focus or Get Better Units) or after an unsuccessful aggressive war.
	The change only affects AI-AI vassal agreements. Could also implement it in a way that makes civs more reluctant to become vassals of human civs, but that's a bit more work, and I think it's still difficult enough for humans to obtain AI vassals because humans can't generally afford large enough armies to impress the AI.
	Probability based on rank should reduce snowball effects.

When a vassal is more advanced than its master, the vassal may decide to gift tech to the master from time to time. This is contingent on a sufficiently high relations value (Cautious when capitulated, otherwise Pleased).	The master may gift techs to the vassal but not vice versa.

Rationale	The master should gradually catch up with a technologically more advanced vassal. Normally this happens through tech trading, but sometimes the vassal is too far ahead. Not (historically) plausible that it takes a civ forever to absorb the tech of its vassal. From the vassal's point of view, one can argue that a voluntary vassal is – to an extent – invested in its master's success. A capitulated vassal prefers to break free, but this is always a long shot and, if relations have thawed, helping the master win is still better than a rival victory.
See also	130v makes vassals always Friendly toward their master – this does not apply when deciding whether to gift tech.
	In this demo game of mine with an earlier version of AdvCiv, Tokugawa managed to make Willem capitulate in the midgame despite being some 10 techs behind; the two remained unable to trade tech for much of the rest of the game because it took Tokugawa too long to catch up.
	130z deals with tech gifted between non-vassal rivals.

AdvCiv	*BBAI* (v0.83)
Disabled BBAI's human-as-vassal option.	This option had allowed human players to offer themselves as vassals to an AI master. The option was disabled by default (through XML). K-Mod enabled the option, but K-Mod AI changes `CvTeamAI::AI_surrenderTrade` seem to have broken it. At least in K-Mod 1.45, I can't bring any AI civ to accept a human vassal, and the explanations (denial text) don't make much sense.
Rationale	Looks like too much work to make this work. Also, it's rarely smart to accept a human vassal – it's not going to send military assistance and will work hard on breaking free.
See also	Rise & Fall and Debug mode allow human players to take control of a vassal. This still works; I've only disabled the option of becoming a vassal through diplomacy.
Config	Can re-enable it through `BBAI_Game_Options_GlobalDefines.xml`, but I don't think the AI is ever going to accept.

112b	AI changes to surrender decision
See also	[123d](#) blocks an exploit where cities gifted to a war ally can lead to faster capitulation.

AdvCiv	*K-Mod*
AI civ doesn't surrender unless there are numerous enemy units in its territory on an important landmass (or having been nuked). Also refuses to surrender If 30% or more of the civ's population are on a landmass with few enemy units.	Enemy positions matter for peace treaties, but surrender is only a matter of power and war success.
~~Response is "You'll have to take it from our cold, dead hands" if denied.~~ "It looks like your offensive has run its course" as of AdvCiv 0.99.	That response isn't used anymore at all. BtS used it for cities that the AI didn't want to trade.

Rationale	Let the enemy demonstrate that they can reach our important cities before capitulating.
See also	[CFC post](#) about the refusing-to-surrender response.

Don't surrender while there are units en route to the master's territory.	This is checked before signing peace treaties but not when considering surrender.
If a war enemy is the worst enemy of an AI civ and the attitude toward that enemy is Furious, the enemy's power is treated as 10% lower than it actually is. I.e. the enemy needs slightly more power in order to achieve capitulation.	The worst enemy's power is treated as 25% lower regardless of attitude.

Rationale	Should perhaps disable this power adjustment entirely. It makes some sense flavorwise, but makes capitulation easier to achieve when the winning side already has a much larger army when the war starts; otherwise, it'll take time to get the upper hand, and "This war spoils our relationship" will typically lead to a Furious attitude. From a human pov, it's annoying if the AI won't capitulate when a war has already become tiresome.
Tbd.	Attitude could play a bigger role when there is more than one powerful war enemy.

When considering capitulation, the power modifier based on war success is applied to the vassal's power, meaning that war success matters not just for the vassal-master power ratio but also when comparing the vassal's power with the average power of other civs.	The master's power is modified based war success, and therefore only matters when comparing the vassal's power with that of the master.
A civ refuses to capitulate unless its power rating is at most 76% of the global average power rating.	The threshold is 80%.
The impact of war success on capitulation is reduced a bit overall. Can reduce the vassal's power by at most 45%.	Can increase the master's power by up to 100%.

Rationale	Perhaps the average power shouldn't matter at all for capitulation. Now matters less when the master's war success is high.

113	AI worker production and worker assignment to landmasses and cities

See also	reduces the AI work rate bonuses from the difficulty setting.

AdvCiv	*BtS*
The City AI prioritizes workers until the total number of needed workers is reached. And several tweaks to the probability of training a worker and the target number of workers (total and per city).	Workers prioritized only when there are far too few of them; otherwise, any decent building takes precedence.

Rationale	The K-Mod AI tends to train 1 worker per city (BtS: fewer), which is about right for an experienced human player, but the AI isn't good at scheduling its workers, so 1/city isn't enough. Also note that the AI generally isn't good at choosing city production. Can't go too wrong with a worker – except in the very early game and unless there are evidently too many already. Should be about 1.5/city now. (Also a matter of traits, leader personality – especially `BuildUnitProb` –, coastal cities, colonies ... so it varies a lot.)
Config	Can be tweaked through `WORKER_RESERVE_PERCENT` in `GlobalDefines_advc.xml`, but that only affects the upper limit for the total number of workers that an AI civ may produce and not (or only indirectly) the number of workers assigned per city and how worker production is prioritized.
Tbd.	Should take into account currently researched tech (e.g. Calendar, Railroad). Workers without any important task should flock to a settler (if any) or even start building a road toward any guarded city sites (`CvUnitAI::AI_guardCitySite`); will currently only move there once a new city has been founded (except for colonies: workers are already being shipped together with settlers).
See also	: AI chopping; : Forts; : Improvements on landmasses without cities Fuyu has tried to get the AI to build more Workers as well, but I think, by now, my own code supersedes his; haven't looked at the details though. The code in question is in `CvCityAI.cpp`, marked with "Build more Workers" (also "Worker Counting"?).

The number of Workers that the AI aims at having near a city takes into account Forests and Jungles on workable tiles.	The AI only checks for unimproved tiles. Whether the missing improvements will require extra work for clearing terrain features isn't checked.

Made the AI more reluctant to chop Forests near a city that is producing a Worker.	

Rationale	If the AI has time to chop, then it can't be that short on Workers.

When a city has 2 assigned workers but requires only 1, the AI considers assigning one of the workers to another city.	Not sure if BtS can reassign workers at all so long as there is anything left to improve near the current city. The code for reassignment is apparently erroneous (K-Mod comment: "*Is it just me, or did they get this backwards?*") The K-Mod code still got it somewhat backwards I think (see my own comment in `CvUnitAI::AI_workerMove`) and doesn't allow reassignment when a city has 2 workers and needs only 1.

Rationale	This change/ bugfix should make AI worker management quite a bit less inefficient. A BtS comment (by Blake probably) above `CvCityAI::AI_getWorkersHave` says: "*Workers have/needed is not intended to be a strict target but rather an indication. if needed is at least 1 that means a worker will be doing something useful*" So, the number of workers needed is a conservative estimate; more workers than that are probably way too many.

When deciding whether to let a small city grow before a settler or worker, the AI anticipates border expansion and improvements.	The decision is, in part, based on the yields of the tile that an additional citizen would work. Unowned tiles are disregarded and only existing tile improvements count.
	BtS has an extra clause that prevents cities below size 3 from producing any workers or settlers until the owner has at least 3 cities. K-Mod and AdvCiv have mostly removed this restriction.
Rationale	Especially relevant for the capital at game start.
See also	A bugfix in the tile evaluation code (`CvCityAI::AI_getPlotMagicValue`) has been tagged with "advc.001".
Made the AI reluctant to produce Workers in colonies that had already been fully improved at an earlier time and need more Workers only due to a new tech or a change in AI yield weights.	The number of other cities on the same landmass isn't taken into account when the AI decides whether to produce a Worker.
Added code to avoid shipping out more Workers than a landmass can spare.	All idle Workers can get shipped out and are then urgently replaced because BBAI has added some high-priority clauses for landmasses with 0 available Workers.
Fixed a likely bug in the code for ferrying Workers in between cities: Only the Workers needed by each potential target city had been counted, not the workers already available there.	The number of needed Workers decreases as improvements are built, so the error is self-correcting. However, in the late game, the AI may quickly ferry all idle Workers to some small landmass and delete most of them once they become idle again because the landmass will then have far more Workers than needed.
Rationale	These changes were prompted by a test on an Archipelago map with Tropical climate starting in the Modern era. The AI ended up deleting dozens of Workers. Deleting some Workers is reasonable under these settings – Workers are relatively cheap this late in the game and a colony (with free initial population due to the start era) on a large Jungle-covered island needs a large group of Workers a.s.a.p. However, the K-Mod code kept producing, ferrying and deleting Workers even once all land was settled and improved.

113b	Changes to the computation of workers available to a city
AdvCiv	*BtS*
Renamed `CvCityAI::AI_updateWorkersNeededHere` to `AI_updateWorkersHaveAndNeeded` and made that function more efficient.	BtS comment: "*How slow is this? It could be almost NUM_CITY_PLOT* [i.e. 20] *times faster by iterating groups*"
Rationale	A profiler run suggests that this change might reduce late-game turn times by 1 to 1.5%.
AdvCiv	*K-Mod*

Idle Workers that have retreated to a city tile are counted as available to that city; Workers in cargo destined for another landmass are not counted as available.	Workers that have retreated to a city (which is what idle Workers normally do) are not counted. Workers in cargo are counted regardless of the transport's destination. The BtS code had worked differently, but probably also incorrectly.
Rationale	A bugfix really.
AdvCiv	*BtS*
The number of Workers available to a city is updated over the course of a turn.	There's an update once per turn (still true in AdvCiv); updates throughout a turn happen in `CvUnitAI::AI_improveLocalPlot` and `AI_nextCityToImprove`, which covers only a fraction of the possible Worker moves.
Rationale	Mainly to avoid transporting or deleting too many Workers in one turn.

114	Changes to AI attack courage
114a	Breaking sea blockades
AdvCiv	*BBAI*
AI willing to break a sea blockade using inferior ships. Will attack at near-0 odds if the defenders are sufficiently outnumbered.	AI keeps building e.g. Triremes but never dares to attack even a single Privateer. (Not sure if it would attack with Caravels.)
Rationale	Known issue in BBAI, not fixed by K-Mod. I've also posted about this on CFC, but the code I posted there is now outdated; doesn't fully solve the problem.
Tbd.	Would probably be smarter to stop building primitive ships, and wait for better tech. Also dubious that there is dedicated code for breaking blockades – why should the normal AI behavior for stack combat be inadequate for this?
114b	Attacks on valuable units at poor odds
AdvCiv	*K-Mod/ Lead From Behind*
Changed the attack courage computation so that the cost of the involved units is given less weight when the odds are one-sided.	AI leaders and the Barbarians calculate with optimistically increased attack odds. Based on these inflated odds, K-Mod does a kind of expected value computation: attacking a Tank with a Warrior at 5% odds is wise because the Tank is much more expensive than the Warrior. (Of course, the true odds are much lower.)
Rationale	Warrior against Tank doesn't matter much, but K-Mod Barbarians are also too happy to attack advanced units in fortified positions.
See also	koshling has addressed this by making the attack odds increase multiplicative; see this revision of "RoM: A New Dawn".
114c	*Currently unused*
114d	Decreased random portion of attack courage

The random summand added to AI attack courage is between 0 and 12 for AI civs and between 0 and 27 for Barbarians. Increased the base attack odds change of Barbarians by 2.	AI units have a bias toward attacking based on leader personality. The bias is composed of a deterministic base value between 0 (e.g. Ashoka, Barbarians) and 6 (Ragnar and Napoleon) and a portion chosen at random once per turn between 0 and 32 for the Barbarians and between 0 and 16 for all civ leaders.
Rationale	If the AI attacks completely rationally, defensive tactics won't really work anymore; but I think making the AI a bit more rational won't hurt.
Config	Partly through the DLL, partly through `Civ4LeaderHeadInfos.xml`.
114e	City attackers in a friendly city may opportunistically attack enemy units approaching the city ("leave attack").
Credits	More Naval AI (Tholal)

115	AI less willing to commit to victory strategies
See also	112: AI civs less willing to become vassals. 018: AI uses Crush strategy less. 019 makes the AI a bit less inclined to use military strategies in Aggressive AI mode. UWAI bases some AI decisions on the victory stage of other civs; e.g. wars started in order to thwart the victory of a rival.

AdvCiv	*K-Mod*
In a game with 7 civs, the AI enters stage 3 (of 4) of the Domination victory strategy when meeting 55% of the (land and population) requirements. More generally, the target percentage for stage 3 is 62 minus the number of civs, and 87 minus the number of civs for stage 4.	The thresholds are 50% for stage 3 and 80% for stage 4, and don't depend on the number of civs. However, the requirements themselves do depend on the number of civs (e.g. 64% with 7 civs, 51% with 16). Victory strategies with stages 1 to 4 were introduced by BBAI. At stage 4, victory is imminent.

Rationale	The K-Mod AI goes for military victories too often (or early) for my taste. The BBAI approach of letting the AI play more rationally is fine, but let's not turn it into a wannabe HoF player.
	50% means 24% of the world population and 32% of the land, i.e. about a "double share" in a standard game: 2 in 7 is 28.5%. This shouldn't quite be enough to trigger domination 3.
	55% means that 35% of the land is needed for stage 3. That's right between 3 in 7 (42.8%) and 2 in 7.
	I'm factoring in the number of civs because it is easier to conquer more land when that land is divided among several weak opponents than when it is owned by a few powerful ones.

Added a condition that makes the AI less willing to go for Conquest victory if there are many rivals on other continents.	No such condition.
Escalated the conditions so that the conquest stage is essentially one less than in BBAI/K-Mod. Added requirements for stage 4: half of the initial rivals, rounded down, need to be defeated.	The half-rivals-defeated condition is checked for stage 4 (which becomes stage 3 in AdvCiv), but there are also alternative conditions for that stage.
Rationale	As above: military victories are too commonly pursued. Also: It's problematic when an AI civ goes for a military victory before naval invasions become feasible, because it's too difficult for players on other continents to interfere in time.
Made the conditions for culture victory strategies more narrow.	
Rationale	Culture-loving AIs can be unpleasant to play against (culture pressure, wonder grabbing), so the AI should only go for it when there is a realistic chance of success.
(Only relevant if UWAI is disabled.) When in multiple wars at once, the AI ignores its military victory stage and applies the normal decision process to any war that is no longer recent. I.e. multiple wars are still possible, but the AI won't just refuse to talk.	AI in Conquest 4 or Domination 4 never ends a chosen war so long as war successes are favorable.
Rationale	It's OK that an AI close to a military victory likes to fight wars, but it doesn't have to insist on fighting everyone at once.
Conquest 4 only when owning at least half of the world's cities.	
Rationale	Even when militarily superior, conquering so many cities is going to take time.
Space victory not pursued if total production clearly insufficient. Stages 3 and 4 require a significant portion of the necessary parts to be completed.	Once Apollo Project is built, stages 3 and 4 can be reached just through technological progress.
Rationale	To discourage small civs, say, with just three cities, from pursuing a space victory. If they want to have a chance, they'll need to expand instead. Also to discourage civs that had come close and then lost half of their cities at war; such civs should at most be at stage 3.
If a civ has a very large total production rate in the endgame, Space victory is considered even if other civs are a bit more advanced technologically.	
Rationale	Mostly to allow civs that are pursuing a military victory to switch to Space victory if they run into a stalemate with another very powerful civ.
Domination 2 requires the AI to be in the upper third of the scoreboard.	Upper half suffices.
Rationale	Military victories can't really be won from way behind. (Which isn't to say that civs in the middle of the scoreboard shouldn't start wars.)
115b	Stages for Diplo victory revised
AdvCiv	*BBAI/K-Mod*

Based mostly on the current (voting) population of the own team, vassals and friends. Personality and randomness still factor in; game options don't (unless Diplo victory disabled).	Based on leader personality, randomness and, a little bit, on Aggressive AI and Always Peace game options.
Stages 3 and 4 can be run regardless of other victory stages.	Stages 3 and 4 are impossible if already in stage 3 or 4 of another victory condition.
AI prioritizes food a bit at stage 4 (grow votes).	K-Mod takes into account the favorite civics of rivals when evaluating civics at stage 3.

Rationale	This part of the victory-stage code hadn't been revised by karadoc and it didn't do anything intelligent. Checking for peaceful game options isn't good: Diplo victories are usually half conquest, half diplomacy.
	Diplo victory isn't often a viable route for the AI. I've made the changes mostly because I had already written code for estimating voting populations for UWAI.

Diplo victory stage 2 factors into the decision to build AP and UN. The AI avoids building AP/UN if another civ seems closer to Diplo victory.	Only stage 1 is considered for AP/UN. No avoidance. (NB: Building evaluation also affects AI tech paths, i.e. the AI is able to aim at Mass Media – if it can properly evaluate the UN.)

Tbd.	Doesn't look reliable; at best, it'll no longer build the UN when it's *obviously* a terrible blunder.
See also	178 is supposed to improve CityAI decisions about the AP. 031 makes the AI more willing to found cities when near Domination.

AdvCiv	*BtS*
When near a religious victory, the AI is more inclined to pursue the "Missionary" strategy, and its vassals switch out of Theocracy.	"Missionary" is a BtS strategy that is adopted based on AI flavor and the number of civs with Open Borders and whether they've already converted. Victory doesn't play a role.
	The AI makes no effort to deal with Theocracy.

Rationale	Should at least allow the AI to win an AP victory when all human players are already members of the AP. Will otherwise have to use Holy Wars to bring down the humans.
Tbd.	AP victory conditions need to be overhauled. It's silly that a single converted city makes all the difference in victory votes. (War votes are already addressed by kekm.25.)

The AI proposes the victory resolution when it gets the chance and a team member is at Diplo victory stage 4.	AI chooses uniformly at random from all the resolutions that it supports.
Voting AI abstains if it likes two candidates equally.	Votes for the candidate with the lowest internal id; normally humans have lower ids than AI civs.
Also abstains when at stage 4 of a victory condition.	May vote for someone else's diplo victory even e.g. after launching a spaceship.

Tbd.	The AI still proposes random resolutions in all other cases, and this is often agonizing to watch.
	Under "let's discuss something else", should allow asking the AI whether it would vote for the active player. Could call the option sth. like "If there was an election next tuesday ..." Then just call `CvPlayerAI::AI_diploVote` (should make that `const` too). Snarky attitude-flavored responses would be nice.

115c	Victory strategies in zero-sum games
AI doesn't pursue Diplo victory when there's just one other team left; AI does pursue Conquest if there is just one other team to begin with.	Diplo2 possible with just one rival, but UN and AP require at least two other teams. Conquest1 requires at least one civ to be a vassal or eliminated; doesn't work if there are only two teams at game start.
Rationale	Both only really relevant for games against a single AI opponent.
115d	Show human victory stage in debug mode
Debug mode has no bearing on victory stages. (No change.)	The victory stage of human civs is computed as if they were AI civs when in debug mode. Also, human civs can never be at stage 1 or 2 of a victory strategy, only 0, 3 or 4.
Rationale	Since [R&F](), [UWAI]() and other parts of the AdvCiv AI code use victory stage in order to determine whether a (rival) civ is getting close to a victory condition, it's important to see the human victory stage the same way that the AI sees it. Stage 1 and 2 are probably too unreliable and might lead to peculiar decisions from AI governors of human cities. I'm guessing that's why BBAI restricts human victory stages. Fair enough; the AI only needs to know when a human is close to victory; 0, 1 and 2 don't (need to) make a difference.
Tbd.	The `AI_calculate..VictoryStage` functions aren't ideal for determining how close a civ is to victory. The main purpose of these functions is to determine whether and how much the AI should focus on a particular victory condition; therefore, e.g. AI flavor values factor in. Would be better to write separate code for measuring the progress toward victory.
115e	Minor misc. changes to AI decisions about victory strategies

115f	AI redistributes personality-based weights assigned to disabled victory conditions
AdvCiv	*BBAI*
When a victory condition is disabled, any AI weights assigned to that condition are redistributed among any valid victory conditions that already have a positive weight. The redistributed weights also affect AI decisions about the use of Great People. Some other AI decisions – which seem to interpret the victory weights as more general personality traits – still use the weights as defined in XML.	For each AI leader, a weight value is defined in XML for every victory condition except Time. The weights give the leader personal preferences for some of the victory conditions. When a victory condition is disabled, the preferences for the remaining conditions do not increase. E.g. when Space victory is disabled, AI leaders become more interested in Culture only insofar that being well positioned for a Space victory can't get in the way of plans for a Culture victory.
Rationale	Don't want AI leaders to play (somewhat) aimlessly when some conditions are disabled. However, zero weight should still mean that the AI doesn't pursue the respective victory condition.

116	Changes to raze decisions
See also	[250b]() moves the No City Razing option to the bottom of the Custom Game screen. [ctr]() makes the AI raze cities in awful sites. [300]() deals with razing by Barbarians.

	[cdtw.1](#): razing by vassals.

Tbd.	Try to use `CvPlayerAI::AI_assetVal`.

AdvCiv	*K-Mod 1.44*
If the AI raze value plus a random number between 0 and 5 is above 0, the city is razed. 0 to 5 is very little; it's hardly random at all.	Raze value is used as the probability of razing a city, i.e. even if the value is just 1, the city might be razed (with probability 1%).
The AI leader's `RazeCityProb` adds at most 15 to the raze value. Settled Great People count as 5 against razing.	`RazeCityProb` adds up to 75 to raze value. Settled GP count as 2 against.
Impact of distance and finances lowered; impact of distance adjusted to map size.	Cities past a distance threshold are usually razed.
Unlikely to raze when controlling fewer than 5 cities, i.e. in the early game.	Cities conquered in a very early rush are usually razed because of the distance.
When no other city on the continent is owned by the conquering AI civ or when the city is very far away from that civ's territory, the AI tries to predict if other cities in the vicinity are going to be conquered. If so, the city is less likely to be razed.	K-Mod has a similar clause, but it only checks if the war plan type is "total".
	(Some improvements in K-Mod 1.45, but my changes are more comprehensive.)
Rationale	K-Mod razes too much and too randomly, and this is among the most common complaints about K-Mod. Leader personality has far too much impact: Genghis Khan has 75 RazeCityProb while some leaders have 0. A city would have to have e.g. 15 settled GP to make up for that difference (assuming each GP counts as 5; in K-Mod it's actually only 2), or 5 active wonders (each counts as 15).

AdvCiv	*BtS*
AI factors cultural ownership into raze decisions. Reluctant to raze cities with own majority culture or with majority culture of a partner civ (i.e. attitude Pleased or higher).	Tile culture is ignored. The AI incurs diplo penalties by razing cities with cultural majority of a third party.
Rationale	The Diplo penalty is usually not worth it. Can give the city away if it becomes too costly. (The AI knows how to do that too.)

AdvCiv	*BBAI*
When an AI civ conquers a city that, if reconquered, may soon lead to a Culture victory of the previous owner, the conquering civ razes the city if reconquest seems plausible (based on power ratios and nearby units).	The dangerous city is razed in any case.
Rationale	The BBAI comment actually said to raze unless we "overpower" them, but no power check was there.

AdvCiv	*K-Mod*
The AI is extra reluctant to raze Barbarian cities.	Some parts of the raze value computation already directly or indirectly take into account whether the city is Barbarian.
Rationale	Some advantages of conquerting cities from Barbarians probably aren't fully covered by the K-Mod code. E.g. they tend to have smaller tile culture values and other Barbarian cities hardly exert any culture pressure. Also, it practically never makes

	sense to raze a city just to make sure that the Barbarians can't ever get it back.
	More importantly, the AI doesn't currently evaluate whether Barbarian cities are worth conquering (or if it does, it's not working well). When a city is razed, this AI flaw becomes obvious to the human player. Better try to cover that up than to break immersion.
See also	*Tbd.*-note toward the end of change 300 about improving the AI for targeting Barbarian cities.

117	AI chops more Forests	
AdvCiv		*K-Mod*
AI is always somewhat willing to chop depending on available Workers and competing Worker tasks. More specifically, assigns a priority of 0.5 * P in situations where the K-Mod AI is unwilling to chop, and 1.5 * P otherwise. When computing the Workers needed at a city, chopping opportunities are taken into account.		Only chops while building something urgent, or a building in a small city. Then assigns a priority P that is proportional to the chopping yield. Chopping seems to happen only opportunistically, i.e. when a Worker is assigned to a city in order to build improvements, it may (afterwards) be instructed to chop if there happen to be Forests.
Rationale	K-Mod tries to discourage extensive chopping, but I don't think this can work without reducing or delaying chopping yields further (beyond what patch 1.61 did). As it stands, the K-Mod AI is missing out on early chopping yields. See also this discussion on CFC. In particular, there are few reasons not to cut down Forests along rivers and on hills. With Replaceable Parts, a hill Lumbermill yields only one more commerce than a Mine, and, in part due to change 902, a river Lumbermill can't compete with Watermill. As a result of my changes, AI Lumbermills on hills/ at rivers don't seem to occur anymore at all. The other Forests remain largely intact.	
Tbd.	Chopping along rivers is historically sound but on hills not so much. Perhaps I'll allow Mines to coexist with Forest (though some areas were deforested for producing charcoal for smelting ores). The AI should arguably chop even more. I intend to nerf chopping instead, probably by restricting the yield to apply only to buildings and ships. Should arguably only chop if that'll speed up the current city production, i.e. if it isn't already about to finish. I've only implemented that for the decision to chop while producing a Worker (covered by 113) so far.	
See also	113 also includes chopping opportunities in the estimated Worker tasks. 064b prevents chopping production from being converted into overflow gold.	

AI chops Forests outside of city radii. Restrictions:	The AI never chops Forests on tiles that no city can work on.
• Only when there is nothing else to do.	
• Not when automated and chopping is disallowed in options.	
• Not if there is already anger from global warming.	
• Not near planned city sites.	
• Not on improved tiles (e.g. Fort).	
• The AI builds a route to the tile before chopping.	
Possible future cities, distance and the correct timing of the chop aren't considered.	

Rationale	My conditions are simplistic and rather conservative. At least the AI now fells unworked Forests within its own borders at some point.
Tbd.	Considering to set 0 yield from chopping outside the BFC; then this change will be obsolete.
See also	[119](#) prohibits chopping on unowned tiles, i.e. the AI doesn't have to worry about those tiles. [012](#) gives Forest and Jungle a defensive bonus that only the tile owner benefits from. Therefore [this](#) change in K-Mod 1.45, which makes the AI more inclined to chop Forests in the inner city ring, has no effect unless change 012 is disabled. (I've still merged that K-Mod change.)

118	AI changes regarding peacekeeping and city votes	
AdvCiv		*K-Mod*
AI proposes and votes for peace if it likes both sides and neither side is clearly winning or losing. "Like" means that the attitude is *strictly greater* than `DeclareWarThemRefuse-AttitudeThreshold`.		AI only seeks peace if it likes the losing side or dislikes the winning side. "Like" means attitude greater than or *equal to* `DeclareWarThemRefuseAttitudeThreshold`.
Rationale	A minor thing that vexed me in one game. The AI should stop inconclusive wars between its partners. The second part is probably a bug in K-Mod. A comment says, "if [we] like them enough to not declare war on them", and this isn't what the code does.	
See also	UWAI ([104n](#)) partly handles peace vote decisions, but 118 still applies.	
An AI civ will always defy when its last city is to be assigned to another civ.		1 chance in 3 to defy when a city owned by the AI civ or its teammates is to be reassigned. That's all.
Credits	krikav managed to eliminate an AI civ through an Apostolic Palace vote: CFC [link](#)	

Tbd.	Check how important the city is relative to the AI civ's total economy and how much defiance will hurt etc. BBAI comment: "*Wonders, holy city, aggressive AI?*" Ideally use existing city evaluation functions.

119	Can't chop Forest, Jungle outside borders	
Worker builds that remove features can only be built in plots owned by the Worker's team.		Features can be removed from unowned plots and even from plots owned by a war enemy.
Rationale	Forests being chopped for marginal gains by idle Workers isn't realistic. In part, the problem lies with Workers working for free, but large-scale deforestation without nearby human settlement is strange in itself. Moreover, the AI doesn't chop outside of its borders, and it's easier to change the rules than to change the AI. Part of an overall effort to reduce and delay deforestation (without inhibiting the AI).	
See also	117: AI chopping on owned tiles that aren't workable.	

120	Usability and AI improvements for espionage	
See also	132 makes the AI use the force-religion mission less (also changes the mission cost).	
Rationale	Espionage in BtS is beyond redemption. Focus on the few parts that somewhat work, and try to make the rest easier to ignore.	
AdvCiv		*K-Mod*
Default espionage weight set to 0.		1 in K-Mod (since v1.30), was 0 in BtS.
Rationale	Important in the (frequent) situation where a player sets some espionage weights before meeting all rivals. When meeting another rival, a default weight of 0 means that no points are assigned to that rival until the player readjusts the weights in the espionage screen. 1 could mean that a few points are assigned or a lot, depending on the weights set previously.	
AI less worried (50%) about war opponents having a leg up in espionage. Not likelier to train Spies when fighting a war, only during war preparations.		War plans increase the odds for training a Spy as if running the Espionage Economy strategy.
Rationale	It's better to focus commerce on research or entertainment when at war, and production on military units. Espionage is for cold-war situations. That said, when war is declared, relations take a dive and the enemy may quickly order some malicious Spy missions. It's good to prepare for that, and some extra Spies for lowering city defenses could also be helpful (although the AI rarely manages to use that mission).	
Messages about rival counterespionage missions are shown in white, not in red.		
Rationale	Often, the affected player isn't even using espionage, so a counterespionage mission shouldn't use the color of alarm.	
Capitulated vassal disregard their master and other vassals of the master when checking for tech-steal targets.		Not sure if there's something to prevent capitulated vassals from trying to steal their master's tech (or anyone's).
Rationale	When a vassal switches to an espionage economy to steal tech from the master, then the commerce generated by that vassal is essentially lost to the master. This would be	

	too much of a disadvantage, potentially discouraging players from accepting vassals.
Tbd.	Capitulated vassals should only ever attempt to steal techs that the master doesn't already have.

Made some tweaks that reduce the weight that the AI assigns to espionage commerce (now in a function `CvPlayerAI::AI_calculateEspionageWeight`).	
Rationale	My impression is that, on average, the AI weight should be as defined in `CIV4CommerceInfo.xml`: 1 gold being worth 0.25 espionage. There are situations when espionage is worthless or almost, so the average weight excluding such situations should be a bit higher, somewhere between 0.3 and 0.4. The K-Mod code seemed to end up at the upper margin of that range.
See also	Actually, the average weights assigned by K-Mod were quite a bit higher, but that was due to a bug described under 001.

120b	AI Spies less malicious
Tbd.	Higher AI diplo penalty for malicious missions, and no diplo penalty for failed missions. E.g. 2/3 chance of a diplo penalty (and spy identity revealed) for successful malicious missions, 1/3(4?) for successful non-malicious missions. Espionage screen should then indicate which missions are considered to be malicious.
See also	130v makes Spies less likely to attack capitulated vassals.

AdvCiv	*BBAI*
AI uses "malicious" espionage only when Cautious or Annoyed, depending on the leader's no-war threshold. (If no war at Pleased, then malicious at Annoyed; if no war at Friendly, then malicious at Cautious.) Aggressive AI has no impact on AI malice. (AI may also be malicious when planning war or against a civ that is close to victory; no change.)	Attitude threshold not leader-specific. With Aggressive AI, malicious unless Friendly, otherwise malicious unless at least Pleased.
Rationale	The "take that" missions are rarely beneficial for the AI (although the K-Mod AI uses them quite well). I like them for flavor, but, flavor-wise, malicious espionage only makes sense against enemies. Aggressive AI still has an indirect effect because war plans enable malice, and Aggressive AI leads to more AI war plans.
No fomented unrest, poisoned water or sabotaged building while a city is in disorder.	Cities in disorder are only exempt from sabotaged building K-Mod comment: "*disorder messes up the evaluation of production and of building value*"
Rationale	Cities in disorder won't lose food.
No revolt incited when city defenders are overpowered by a factor of 8:1 or worse.	The K-Mod code only make sure that the defenders aren't too powerful. Will even use the revolt mission against undefended cities. (Whereas BBAI had a clause – disabled in K-Mod – that blocked the revolt mission when the attackers were more than twice as powerful as the defenders.)
Rationale	2:1 as in BBAI would be too strict. Just because a city can be conquered without inciting a revolt doesn't mean that a revolt isn't worthwhile.
The Espionage Economy strategy greatly reduces AI espionage weights against civs from	K-Mod added this strategy; comment in `AI_Defines.h`: "*run high espionage slider to*

whom no techs could be stolen.	*steal techs at a discount.*"
	When hoping to steal a tech from a particular civ, the weights of other civs are halved.
Decreased chance malicious missions further when in Espionage Economy.	Espionage Economy already decreases the chance of malicious missions.
Rationale	Espionage Economy tends leads to a much higher espionage output than normal. Even 10% of that can amount to a lot of wasted commerce if it's spent on frivolous missions.

120c	Hide the espionage slider when it's at 0
AdvCiv	*BtS*
Espionage slider not shown on the main interface when it's at 0. Added the slider to the Espionage screen.	Once Writing is discovered, the Espionage slider is shown on the main interface, city screen and Finance tab of the Economics Advisor.
Rationale	More room on the main interface. Many players hardly ever touch the espionage slider.
Config	Optional through the "General" tab of the BUG menu. Since AdvCiv 0.98, the slider is again always shown by default.
See also	[120g](#) requires Alphabet for adjusting the slider. [History Rewritten](#) also places the espionage slider on the Espionage screen (though I don't suppose it's removed from the main interface).
Tbd.	Occasionally (rarely?), the Espionage screen gets garbled when the slider is moved from 0 to 10. See comment in `CvPlayer::setCommercePercent`.
Position the Domestic Advisor window so that there is space for two sliders above it and space for unit icons below it.	Positioned so that there is room for three sliders; unit icons are half-obscured.
Rationale	Even if the espionage slider is shown on the main interface, I don't think it's important to show it when the the Domestic Advisor is open.

120d	Info on Espionage screen revised
AdvCiv	*BtS*
The heading above the point thresholds for passive espionage is "Threshold".	Says "Cost"; same as the heading for the espionage mission cost.
Thresholds that the player has reached and mission costs that he/she could pay are shown in green.	All in white.
Rationale	If it isn't paid, it isn't a cost.
Credits	The green numbers I've seen in another mod, probably BULL or BUFFY.
No mission costs are listed if the player can't train Spies (i.e. prior to Alphabet). Exception: Costs are shown if the player controls a Great Spy.	Mission costs are listed as soon as the player learns the location of a rival city.
Rationale	Rather distracting to see the costs early in the game.
See also	Ties in with change [004w](#) (uncluttering the UI). A Great Spy prior to Alphabet is a rather academic possibility since the Great Wall no

longer provides Great Spy points (change [310]).	
If mission costs are shown, the Sabotage Production/Improvement/Building missions are shown as a single item "Sabotage" with mission cost "?". Sabotage Project is shown with a cost, but omitted entirely if the city doesn't have a project (a.k.a. Spaceship Part).	Three separate entries. Costs are shown for all except Sabotage Improvement. The cost for Sabotage Production [gives away] the number of hammers spent on the target city's unfinished production. The cost for Sabotage Building is based on the cheapest building in the city.
Rationale	The Sabotage Building cost is mostly unhelpful – unlikely that the player wants to sabotage the cheapest building. The amount of invested production is useful information (close to completing a Wonder?) – but arguably information that the player shouldn't get for free. The cost for Sabotage Project doesn't give anything away I think; Spaceship Parts are listed on the Victory screen.
See also	[103] allows Spy units to investigate cities; that's the fair way to find Wonders under construction. [045] hides buildings in rival cities (meaning that the Sabotage Building cost does give away secret information)
See also	Should show the missions as in BtS when able to investigate the city. And should additionally name the cheapest building and the current production and its progress as "hammers/hammers needed".
"Steal Technology" is shown without a cost if the player is unable to trade techs with the owner of the target city. If they can trade, the cheapest tech (on which the mission cost is based) is shown in parentheses in addition to the cost. "Steal Treasury" shows the amount of gold that would currently be stolen (and on which the shown mission cost is based).	The cost for stealing the target's cheapest tech is shown unless there is nothing to steal. Can compute the amount of gold from the shown mission cost.
Rationale	Don't want to give away the cheapest tech in situations when tech is otherwise secret (pre-Alphabet, No Tech Trading option). If a cost is shown, the cheapest tech can be deduced; more convenient and transparent to just name the tech. Same with gold.
See also	[004i] tells a player from whom gold was stolen how much was stolen.

120e	AI response to poisoned water and unrest
AdvCiv	*BtS*
The effects of poisoned water and fomented unrest don't affect the AI population target, meaning that the AI tends to prioritize food more after a spy attack (in order to keep the current population despite food lost due to anger and bad health).	The effects of spy attacks are treated just like other causes of bad health and anger when setting a population target, meaning that the AI lets its population shrink after a spy attack (may even de-prioritize food).
See also	[160] slows down starvation
Tbd.	The AI should be more upset about poisoned water and fomented unrest than about other spy missions. Could implement that, but spies get identified too rarely (25%; ESPIONAGE_SPY_REVEAL_IDENTITY_PERCENT) for it to matter. Perhaps give these

	missions a 50% chance of revealing spy identity even when the spy isn't caught? See also *Tbd.* under 120b.

120f	Spy missions announced to third parties	
AdvCiv		*BtS*
Spy missions that cause a revolution are announced to third parties that know the mission target. The owner of the spy unit is not named in the notification. Missions that change the religion of a civ are recorded in replays along with the spy owner.		Third parties are notified of revolutions (and they're recorded in replays), but only the target of a spy mission learns about the mission, and no missions are recorded in replays. I'm not sure if the BtS AI uses the revolution missions much; the K-Mod AI does use them from from time to time.
Rationale	It's easy to forget about the possibility of a spy mission and to assume that something is wrong with the AI code when an AI civ switches to a suboptimal civic or religion, sometimes, only to switch back after 5 turns. I don't think there is generally a way to infer whether a revolution was triggered by a spy. A normal revolution causes anarchy, but that anarchy has normally already ended when the next human turn starts. So this change reveals information that players don't have in BtS. I don't like that, but I like an AI that looks incompetent even less.	
Config	Switch in GlobalDefines `_advc.xml` (doesn't affect the replay message though)	

120g	The espionage slider requires Alphabet	
AdvCiv		*BtS*
In games started with AdvCiv 0.95 or later, the espionage slider can't be adjusted until Alphabet.		The espionage slider can be adjusted as soon as another civ is encountered. (This condition is implemented, unusually, in `CvMainInterface.py`.)
Rationale	Don't want to clutter the main interface with this slider long before it is needed.	
See also	120c hides the slider from the main interface when it's 0, but that's optional. And, if 120c is enabled, having an icon for the adjust-slider ability on the tech tree allows me to place a hint in the hover text about the new location of the slider.	
Config	Can be reverted through `CIV4TechInfos.xml` and `CIV4CommerceInfo.xml`.	

120h	Keep espionage-against ratios secret	
AdvCiv		*K-Mod*
Espionage-against ratios aren't shown anywhere (as in K-Mod) and the espionage icon on the scoreboard (disabled by default) is shown when the active player has set a positive weight on the Espionage screen against a rival.		BtS shows as a ratio on the scoreboard and on the Foreign Advisor screen how many espionage points the active player has accumulated against each rival (that info is still available on the Espionage screen) and how many espionage points the rival has accumulated against the active player. Since K-Mod 1.26, this espionage-against ratio is no longer shown. The espionage icon on the scoreboard is disabled by default in K-Mod, but, if enabled, it shows whether the

	espionage-against ratio is greater than 1.
	On the Espionage screen (no change in AdvCiv), an espionage icon is shown next to rivals against whom the active player has set a positive espionage weight.
Rationale	The K-Mod change seems to irk some players greatly, but I don't see why. One reddit user writes that "*it makes espionage much more difficult, annoying and non-worthwhile.*" That sounds like tech stealing strategies depend on the espionage-against ratio somehow, but the cost of espionage missions isn't based on that at all, it's based on the total espionage ratio (points that active player and target respectively have ever accumulated against *anyone*) and that ratio is still shown on the Espionage screen. (Plus, in K-Mod, it's cheaper to conduct missions against civs with a large population.) Also, how hard can it be to guess, approximately, the AI espionage weights?

So I'm keeping the K-Mod change because the espionage-against ratio is misleading (suggests that the ratio of points accumulated against each other is important) and can give away information about unmet rivals and AI war plans. The espionage icon on scoreboard probably can't really give away such information, but it's more straightforward to keep the espionage-against ratio entirely secret. Also, the BtS condition for showing the icon was difficult to guess (as there's also the total espionage ratio shown prominently on the Espionage screen) and nearly impossible to guess when espionage-against ratios are hidden. The new meaning of the scoreboard icon is consistent with the icon shown on the Espionage screen. |
| *Tbd.* | The scoreboard icon probably isn't very useful now. Perhaps add some hover text and/or let a right-click on the icon set the weight to 0. Note that the passive missions (demographics, see research) are already covered by other scoreboard columns (power ratio, current research). |

AdvCiv/ Vanilla Civ 4/ Warlords	*BtS*
Messages about spy missions against a human player are shown at the start of that player's turn (`bForce=false`).	Messages are shown immediately when the missions are executed.
Rationale	Don't want players to infer the spy owner from the timing of the messages. The original Civ 4 code in `CvUnits.cpp` (e.g. `CvUnit::destroy`) had gotten this right; BtS mucked it up.

120i	Steal tech cost based on partial research progress
AdvCiv	*BtS*
The cost for stealing a technology is based on the number of research points that the recipient would need in order to discover the technology, i.e. partial progress counts.	Same, but partial progress is ignored.
Credits	VIP mod
Rationale	More intuitive to me. As for balance, this could make quite a difference for players who never increase the espionage slider. The espionage points generated from buildings alone are usually too few for stealing a technology or, often, carrying out any useful mission. If partial progress counts, then even small amounts of espionage can be converted into research. This will require some attention to detail on the player's part, and that could actually annoy players who dislike the espionage system; but I'd like to

	give it a try.

120j	AI improvements for Spread Culture mission	
AdvCiv		*BtS/K-Mod*
The AI takes into account the amount of city culture that will be spread and also tries to predict how the city tile culture percentage will change.		The BtS code seems to attempt compute some amount of culture, but it actually just uses 5% of the city tile culture, which is not how the Spread Culture mission works at all. The code also doesn't take the total tile culture into account, meaning that the culture added could be a drop in an ocean that won't change the tile culture percentages at all.
See also	A fix for apparent K-Mod bug that had only considered cities previously owned as targets is tagged with change id 001.	
	CFC post by me summarizing how the Spread Culture mission works.	

121	Misc AI changes to Worker builds and citizen assignment	
See also	131: Other minor misc. AI changes 901 tweaks the evaluation of happiness from improvements.	
AdvCiv		*BtS/ K-Mod*
For deciding whether to build a Fort or a cheaper improvement to connect unworkable resources, the AI uses a heuristic that considers the following circumstances: • How busy Workers currently are overall; • whether the tile has natural defenses; • whether a city is planned on or next to the tile; • and whether a Fort would function as a canal.		The Worker AI frequently builds Forts on resources that aren't (yet) workable. Forts are always preferred on these tiles.
AI always prefers any improvement with a positive yield over Forts on workable tiles, and replaces the Fort when a tile with a Fort becomes workable (once a Worker finds time to do it; not necessarily high priority). (Fixed a possible bug that may have prevented improvements that connect a resource – like Forts – from being replaced – even by an improvement that also connects the resource). Probably not a bug after all. I think my change only gives higher priority to replacing Forts.		Once Fort tiles become workable, the AI is hesitant to replace Forts with yield improvements. Also likes to build Farms and Cottages on revealed but yet unusable resources, and doesn't replace them with improvements that connect the resource later on.

Rationale	I've posted some screenshots about the K-Mod problems [here](#).
	Forts cost a lot of Worker turns and often have to be replaced later on, so the AI should be hesitant to build them. Possibly a bug in BtS: The AI picks the most expensive improvement; was perhaps intended to be the cheapest (hard to say).
	My heuristic using natural tile defense leads to a reasonable number of Forts in reasonable locations. Not really smart, but looks good on the surface.
	Forts can theoretically be better than yield improvements even on workable tiles. A Silk Fort preserves the underlying forest, which is worth 1 production vs. the 3 commerce from a Plantation. 3 commerce is generally preferable to 1 production, and I don't think the AI can figure out when 1 production is better. Better stick to the obvious improvements.
Tbd.	I'm still seeing Forts on workable Silk. I intend to add one commerce to Silk Plantation and Fur Camp anyway, will hopefully no longer happen then.
	Could also look into Fuyu's Better BUG AI code marked with "Forts Connect Resources but should never be built"; seems to address the same problem as my code.
	I don't think I've actually seen an AI canal yet (not through `CvPlayerAI::AI_getPlotCanalValue` either).

Increased the value assigned to GPP when choosing jobs for citizens by 38%, but made the value decrease faster with each additional GP.	K-Mod comment: "[...] *because of the flawed way that food is currently evaluated, I need to dilute the value of GPP so that specialists don't get value more highly than food tiles. (I hope to correct this later.)*"
Rationale	I've noticed that the AI doesn't create enough GPs in the first half of the game and rather too many in the second half. I didn't look into the food evaluation issue mentioned by karadoc, but AI cities still seem to grow normally in tests.

AdvCiv	*BtS*
When AI yield priorities are adjusted in response to high expenses, then the adjustment is done gradually over the course of several turns.	The city AI may lower the priority of production yields when (overall) expenses are high. The priorities are fully recomputed each turn without any inertia.
Rationale	Comment in the Dune Wars mod: "*ALN - this causes a few issues in DuneWars and in general. if it causes a wholesale move towards commerce, the ratio changes next turn shifting back to production, leading to a possible endless seesaw [e]ffect*"
	I haven't observed such a seesaw effect, but it sounds plausible. (I didn't adopt any code from Dune Wars; I think their solution is specific to that mod.)

Made the city AI prioritize food more in situations when happiness and health allow for a lot of extra population.

When building "territory" railroads (`AI_routeTerritory(/*bImprovementOnly=*/true)`), the AI prioritizes worked tiles over unworked tiles and up to 3 workers can team up depending on the build time (i.e. mostly on slower game speed settings).	Railroads on worked tiles should arguably be handled primarily by workers assigned to cities, but those workers can't always immediately build all railroads that increase production yields, so workers that improve their owners territory in general end up building such railroads as well. Those workers prioritize tiles solely based on the distance to their current tile. Tiles that another work is already building a route at or adjacent to are disregarded.

Credits	Idea by Elkad ([CFC post](#))

When the AI gains access to a new worker build through a tech trade (or the WorldBuilder), the AI updates the best build and best route-to target stored for each city immediately.	Those AI caches are normally updated at end of turn, which works nicely when a tech is discovered through research. If a tech is received during another player's turn, AI workers will spend a whole turn with cached data that ignores the new build. In the case of railroads, AI workers may begin lengthy "route-territory" missions on that turn, substantially delaying railroads on worked terrain improvements.

Rationale	Mainly for railroads. For other builds, it's normally just a 1-turn delay.

Removed or tweaked various conditions under which AI (and automated) workers had placed routes while moving to a target tile.	In most(?) cases, AI workers headed for a target with the best build value lay down routes connecting their current location to the target tile.

Rationale	The BtS behavior can result in long delays for important builds.
Credits	Elkad pointed out that the BtS behavior is far from optimal when it comes to railroads.
Tbd.	Further improvements regarding railroad constructionsuggested by Elkad: [CFC post](#)

121b	AI uses hurry production less aggressively, especially Slavery
AdvCiv	*K-Mod*

When the production order that is to be hurried is nonurgent, I'm reducing the value counted for overflow production because the subsequent orders are probably also not going to be urgent. Decreased the urgency factor for units so that the AI will mostly only hurry them when there is a war plan. Except Settlers and Workers, which are usually hurried. Buildings that primarily grant happiness and health are only hurried if the city immediately needs happiness or health. And specialist slots are ignored when evaluating the benefit of getting a building earlier (because the population loss will make it difficult to fill those slots). Even when a building has a very high utility and is thus desirable to get earlier, the AI will not hurry the building if that's inefficient in terms of lost food versus gained production – i.e. when the building is a wonder with a hurry penalty. When evaluating the cost of not working a tile due to population loss from Slavery, the value is increased if the tile is unimproved and expected to be improved soon.	K-Mod replaces the lengthy case-by-case BtS code (e.g. "Rush Courthouse if maintenance is at least 10") with a general calculation of lost and gained yields. The benefits of getting a building a couple of turns earlier are evaluated through the same function that is used for choosing production orders (`AI_buildingValue`). For units, it's a matter war plans and Area AI, but cities that can use Slavery efficiently (Granary and low population) will normally hurry units even when there is no war plan, and all buildings too. Wonders are usually hurried despite the penalty because their building values are extraordinarily high. Will hurry too readily in the early game when Workers aren't quick enough to improve tiles. This is a bit of a feedback loop because unimproved worked tiles are a main driver of training additional Workers. Also, in general, if Slavery is used a lot, cities stay small, and thus Slavery remains efficient (small cities need less food to regrow population).

Rationale	The K-Mod yield calculations are very well done I think (certainly better than anything I could write), just the part that evaluates how urgently the city needs production is bare-bones.
	The AI doesn't hurry as much as a highly competent player now, but that makes sense to me because the AI generally produces units and buildings ahead of time (or for no particular purpose) and thus doesn't benefit much from getting them a couple of turns earlier.
	Not sure how much the AI uses rush buying now. (Well, basically, hardly ever as in K-Mod because the AI won't adopt Universal Suffrage, and I can't blame it.)
See also	064b changes the interaction between hurry production, chopping and overflow a bit (rules-wise). 110 shifts yield priorities, in part, to match the use of Slavery. 912d allows Slavery to be disabled for human players.
	Civ 4 Reimagined also tones down K-Mod Slavery: "*AI doesn't use slavery as extensively as before which results in bigger cities and stronger AI overall.*" (source)

advc.ctr	Changes to city trades (change id was originally "advc.122")
Tbd.	Software design: Would be nice to move the AI code for city trades out of CvPlayerAI through object composition and an abstract "TradeAI" class that other trade evaluation classes could be derived from as well in the future, e.g. if the rules for tech trades are changed (that would be under change id 550) so that a rational AI can be put in place. TradeAI should have functions `willGive`, `willReceive`, `giveVal`, `receiveVal`, imposing the structure described under ctr (AI), and should be serializable.
Any non-annual trade items can be offered in exchange for a city.	Cities can be traded only as part of a peace deal; otherwise they have to be gifts.
Rationale	Mainly to make diplomacy more rewarding when tech trading isn't possible, i.e. when playing with the "no tech trading" option, but also when a civ has fallen behind. (I'd also like to move tech trading from Alphabet to Scientific Method and add a tech diffusion system.) Could also make culture-based strategies more rewarding and interactive: Flipping cities through culture often isn't feasible, but a combination of culture pressure and payment can work.
See also	Dawn of Civilization players make a case for city trades here and here. Their arguments seem to be mostly specific to mods based on Rhye's and Fall though.
In order to be able to receive a city through trade, a civ needs to have at least 10 percent tile culture there. Unless at war, the civ also needs to have at least half as much tile culture as the current owner. (Exception: Liberation to colonial vassal.) For trades between human players, the recipient is required to have higher culture if that player has already owned the city at some earlier time and the two players aren't at war.	All cities can be traded between humans. The AI accepts all cities from humans where it has at least 1% culture, and otherwise refuses only if the city is 10 or more tiles away from the nearest AI city, or when in financial trouble (which rarely occurs). Outside of peace negotiations, cities change hands between AI civs only through liberation (which requires much more than 10 percent tile culture). Vassals can only receive cities through liberation; see next blue box.
Config	The 10% are set in `GlobalDefines_advc.xml` (`CITY_TRADE_CULTURE_THRESH`).

Rationale	Should make it harder to gift worthless cities to the AI. It's also implausible that a city could be given over to a nation that has no history there. Tile culture also implies geographical vicinity.
	Ceding a city to a civ whose nationality is clearly a minority there is also implausible – except when negotiating peace terms. Nowadays, ceding even uninhabited territory tends to prompt frantic popular opposition, but this was not so in pre-modern times. Therefore any stricter nationality conditions should be part of the trade denial check so that leader personality can be factored in.
	Generally, only culturally contested cities should be tradeable; too many trade possibilities otherwise I expect.
	The special condition for human-to-human trades is intended as a safeguard against back-and-forth trades (e.g. for bumping units forward).

A master civ can trade any cities to its vassal where the vassal has more culture than the master. The vassal may reject the city.	Can only *liberate* cities to vassals. If a vassal isn't the civ that would receive a city upon liberation, then the vassal can't receive the city at all.

Rationale	I suspect that the restriction was put in place to prevent masters from demanding their vassals' cities, and that the inverse direction had been assumed to be covered by liberation. Not true in the case of a city that is culturally contested between a vassal and a third civ.
	Giving cities to vassals means less human city management; should be encouraged (and allowed, to begin with).
	The stricter culture clause is just flavor. I suppose the master population wouldn't want to be ruled by the vassal.

Vassals can accept cities from rivals, but can't cede cities to rivals, not even through liberation.	Vassals can't cede cities to anyone but can accept cities – except from their master.

Rationale	A vassal gaining extra cities should be fair enough. The master may not like it, but vassal are allowed to work toward independence.
	A vassal giving cities away would be problematic because the master doesn't get to decide that a vassal's remaining cities are no longer worth protecting

See also	[Consequences](#) of a city traded to a vassal: doesn't trigger a peace treaty.

When enemy units (war enemies or Barbarians) are near a city (within a 5x5 square), that city can only be traded to a war enemy. While at war with the previous owner of a city, that city can't be traded to civs that aren't at war with the previous owner. These two restrictions don't apply to liberation. The former is implemented as a trade denial condition with explanation text "We are afraid of your enemies" or (for a human recipient) "it's out of our hands".	Only liberation cares about hostile units, and that clause is only about units hostile to the new owner.

Rationale	To prevent players from giving threatened (and possibly untenable) cities to an uninvolved party. In reality, other civs (including the enemy at the gates) would simply not recognize such a change in ownership.
	Not sure about the liberation exception.

Only revealed cities can be traded for.	Can receive previously unknown cities as part of a peace deal. AI civs accept unknown cities from human civs as gifts.
"... make a trade proposal" option hidden when there are no eligible trade items.	Proposal can lead to an empty Trade Screen.

Rationale	A minor change while I'm at it. Not plausible that the AI is (supposedly) able to decide whether to accept a city that it has no knowledge of. Moreover, unrevealed cities are, in principle, secret in K-Mod, and the Trade screen can leak that info.
	An empty Trade screen is a bit confusing, and became a more common occurrence in the early game after I excluded unrevealed cities. "What do you think of ..." - "Yes?" "Let's discuss something else" is also confusing (when there is no third civ to talk about), and I've rectified that in a different mod (Git commit), but I actually find the BtS behavior too funny to change it.

AdvCiv	*K-Mod*
Apart from colonial vassals, only former owners of a city are eligible for receiving it through liberation.	K-Mod added a "*base culture [value] to dilute the true culture values*." (Git commit) As a result, civs with 0 culture level can be eligible for liberation.

Rationale	I'm not sure if that side-effect of the K-Mod change was intended. I think allowing liberation to arbitrary civs in some obscure circumstances only complicates matters.
Tbd.	The computation of the liberation player is still highly obscure. In summary: Can always liberate to a colonial vassal that has its capital on the same landmass. Otherwise, the liberation player is is the one with the highest score – unless the current owner has the highest score. The score is city culture with a dilution factor from K-Mod, biased for the original owner when liberating upon conquest, strongly biased against rivals of the current owner, taken times a factor between 1 and 2 based on the team' culture in the city tile (K-Mod) and divided by capital distance (air distance, doubled if colony).
	Maybe do something based primarily on tile culture (city culture of other civs isn't even visible on the UI). That already accounts for distances – though nearby cities may have recently changed hands, so a distance factor or sanity check would still be needed.

AdvCiv	*BtS*

AdvCiv	BtS
Keep the option of liberating a city right after conquest. `canTradeItem` is now called to make sure that the (implicit) city trade is legal. Cities ceded this way count as liberated for the "you liberated our cities", "traded fair and forthright" and "traded with our worst enemies" relations modifiers. The raze popup says "liberate" in parentheses to clarify this. The popup says "let the … empire assume control" if the new owner hasn't owned the city before. (This can only happen with colonial vassals; see the previous box.) When the liberation player upon conquest differs from the peacetime liberation player, the popup choice of keeping the city is followed by a note saying that liberation to the original owner won't be possible at a later time.	After conquering a city that can be liberated, the raze popup includes an option to liberate the city if the player that conquered the city is able to enter the territory of the liberation player. The liberation player is computed with a bias for the city's original owner (i.e. the founder). However, the city counts for "you liberated our cities" only if the liberation player would be the same without the bias for the original owner. It never counts for fair trade and enemy trade. The raze popup says "return control to the … empire" even if that player has never owned the city. The recipient can't refuse. (no change) The option to cede a city after conquest was already present in Vanilla Civ 4; when BtS added the notion of a liberation player, the code was updated to cede to the liberation player instead of the highest-culture player.
Rationale	I haven't added a trade denial check because, otherwise, for equal treatment of human and AI, human players would have to receive a popup allowing them to reject a city; way too much work, and it's fair enough to say that the new owner is somehow taken by surprise. The bias for the original owner is an obscure rule – until I read the code, I had assumed that liberation through the raze popup worked just like through the trade screen. That said, it's flavorful to restore a city to its original owner after/ as part of its "liberation." (Perhaps there should be some simpler advantage to it.) The relations boost from liberation has to apply in any case; otherwise, the raze popup is a trap. The trade modifiers aren't all-upside, but treating them differently from liberation would add to the confusion and getting around the enemy trade penalty this way also wouldn't be particularly plausible.
Tbd.	Perhaps remove the bias for the original owner (`bConquest` in `CvCity::getLiberationPlayer`).
~~Can liberate a city even if there is an enemy sea unit or non-combat unit visible from the city.~~	There must not be any unit hostile to the prospective city owner visible from the city. Otherwise, the city can still be traded, but it doesn't count as a liberation (no diplo bonus).
Rationale	The visible-enemy restriction is reasonable – if the new owner has to fight for the city, it's not really liberated –, but units that can't attack the city shouldn't block liberation. Disabled this in AdvCiv 0.97 because trade denial handles hostile units now. That said, the trade denial check ignores all ships. I've left the old code as a comment in `CvCity::getLiberationPlayer` in case that I want to merge it into the denial check. (Removed that comment too on 22 Mar 2020.)

ctr (cont.)	Consequences of city trades
AdvCiv	*BtS*

When a city is traded, the tile culture of the old owner is decreased in the full city radius (21 tiles), except for tiles contested by other cities that don't belong to the new owner. Those rival-contested tiles are affected by the trade if and only if the traded city has a higher priority based on distance and (tie-breaker) age.	The old owner's tile culture is set to 0 in the city tile and in the inner circle. The outer circle remains unchanged.
The tile culture of the new owner is increased by the same amount that the old owner's culture is decreased. The amount of converted culture is equal to 50% of the old owner's culture or 100% of the new owner's culture, whichever is smaller. I.e. the new owner's culture can at most triple, and the old owner's culture can at most halve. 33% of the old owner's city culture is converted into culture of the new owner. This doesn't increase the city's total culture (sum over all civs), so it should be safe in multiplayer.	The new owner doesn't gain any tile culture. If the city is liberated, then the new owner's city culture (not: city tile culture) is increased by 50% of the old owner's city culture. The old owner doesn't lose any city culture. K-Mod disables the culture increase when the new owner had already owned the city at some earlier time; this was done to block an exploit in multiplayer mode. City trades that don't count as liberation don't affect city culture.

Rationale	The instant removal of all culture in BtS is jarring. That said, the mutual agreement should have some (appeasing) effect on the population, so I'm halving the removed culture. Excluding the outer ring can lead to strange borderlines with foreign enclaves. The culture is converted to make the city suffer less from culture pressure, to account for the consensual change in ownership (as opposed to violent conquest), to avoid strengthening the influence of third parties, and because culture shouldn't just vanish. The converted city culture can't be based on the new owner's city culture because the new owner doesn't normally have any city culture. Therefore, I'm not using the tile culture conversion formula for city culture.
Config	`CITY_TRADE_CULTURE_THRESH` in `GlobalDefines_advc.xml`
See also	[kekm.23](#) removes the distinction between liberation and regular city trade when it comes to city culture. Also removes the free defenders that cities used to receive when traded to a vassal.

When a city is ceded, reset the anger timers for Slavery, Drafting and defied resolutions unless both the old and new owner are human or the new owner is human and the old owner an AI teammate.	The timers are reset only if the new owner is an AI civ (and always after conquest; no change).

Rationale	Don't want to treat human and AI civs differently without a good reason, and don't want players to be confused by "We can't forget your cruel oppression" anger from actions of the former owner. Two human players could collude to trade cities back and forth in order to reset anger timers. That needs to be prevented. I don't think this trick will work with a human player and an AI teammate, but let's make sure (I guess).

When a city under occupation is traded, occupation status is set before bumping out any units.	Units are bumped before setting the occupation timer, meaning that units in surrounding tiles are teleported away although a city under occupation doesn't have a culture range.

Rationale	Arguably a bug.

After a non-liberation city trade between non-vassals, a peace treaty is signed automatically.	No peace treaty (unless the city was granted to a human player as a gift or tribute).
Rationale	Don't want human players to sell a city to the AI and immediately declare war and reclaim the defenseless city. Not a problem in BtS because the AI won't pay for cities.
	I want vassals at least to be able to accept cities from rivals, but they mustn't force a peace treaty on the master. A human selling cities to a vassal before a declaration of war doesn't seem like much of a problem; the vassal isn't going to have much to trade.
See also	[146](#) signs a peace treaty after a war trade.

ctr (cont.)	UI support for city trades	
AdvCiv		*BtS*
Announce traded cities on-screen and in replays.		Only liberated cities are announced.
Tbd.	Would be nice to announce the terms of the trade, like it's done for reparations (cf. change [039](#)). Not so easy to do; currently, the city-ceded announcement comes from `CvPlayer::acquireCity`.	
When a city is put on the trade table at peacetime, a peace treaty is automatically added to the trade table as well (and can only be removed by removing the city item).		No way put a peace treaty on the trade table at peacetime.
Rationale	To make sure that players are aware of the implied peace treaty.	
See also	Same for war trades ([146](#)). [104m](#) shows a peace treaty on the trade table when the AI asks for help or tribute.	
	Related [CFC post](#). The technical difficulties mentioned are resolved now. I had simply forgotten to set the peace treaty item to `bOffering=true`. Overall, this was more tedious than difficult to implement once I realized that the offer lists (despite being `const` parameters) can be modified in `CvPlayer::updateTradeList` at the appropriate time.	
New tab "Cities" on the Foreign Advisor screen. Shows a table with all cities that the active player can trade much like the "Tech" tab does for technologies. An occupation (fist) icon indicates when a city can be liberated. When an AI civ is unwilling to cede or receive a city, hover text shows the denial reason. When not all cities fit into a table cell, points of ellipsis are shown and hovering on those points shows the omitted cities.		
To implement the city table, BUG's IconGrid module has been extended, allowing multiple text elements (with hover text and on-click action) to be added to a single table cell.		
Config	The "Cities" tab can be disabled through the "Advisors" tab of the BUG menu.	
Rationale	A bigger role for city trades requires better UI support. The only place that dealt with city trades was the "Liberation" column of the Domestic Advisor (more about that below); not the right place for information about trades, and the Domestic Advisor was already too cluttered.	
Tbd.	Perhaps it's habit-breaking that the "Tech" tab is no longer right next to the "Exit" button. That could easily be amended by switching the "Tech" and "Cities" tab.	
	The BUG option to disable the "Cities" tab could later be removed to unclutter the BUG menu. But so long as city trades are novel, it makes sense to have an option.	
Clicking on the name of one of the active player's cities opens the city screen behind the Foreign Advisor and highlights the city on the minimap. Clicking on the same city name again or right-clicking on any city name closes the city screen.		

Rationale	Want to help player locate the cities. Unfortunately, the camera can't move (ensured by the EXE I think) while an Advisor screen is open. The Foreign Advisor obscures all the interesting parts of the city screen, so the city screen in the background isn't helpful. However, as a side-effect, the city gets highlighted on the minimap (visibility circle and larger city blot) and the player can close the Foreign Advisor to inspect the city and can then close the city screen with right-click (see [004t](#)) to see the city on the main map.
Tbd.	Clicking a foreign city's name does nothing; should at least highlight that city on the minimap, but I'm not sure how to do that (without opening the city screen, which can't be allowed for foreign cities). Perhaps clicking a city owned by the active player should close the Foreign Advisor screen and then center the camera on the city (like the BUG Wonders tab). And right-click for the current behavior (opening the city screen in the background). An alternative approach would be to show just a near-fullscreen minimap on the "Cities" tab with cities that can be traded highlighted through colored blots; all other info in hover text – but could blots on that map have hover text? Also sounds like too much work. Though a nice large minimap could also be useful for … well, screenshots?

The rightmost column of the (non-customizable) Domestic Advisor shows the revolt probability of cities with a positive revolt probability. The occupation (fist) icon is used as the heading for that column.	The rightmost column instead shows a fist icon for all cities that can be liberated to an existing civ. (Prior to AdvCiv 0.97, the icon was also shown next to cities that could be split off as an independent colony.)

Rationale	Liberation to existing civs is now handled by the "Cities" tab. Liberation to a new civ is presented better by the "Liberate" button on the Domestic Advisor screen. It wouldn't hurt to show that information redundantly, but it's not really within the purview of the Domestic Advisor, the icons are pretty obscure and revolt probabilities are more useful (they also require a bit more space, but everything still fits even at screen width 1024 pixels).
See also	The [Immortal Culture](#) changes make revolts more relevant than in BtS.
Tbd.	The Domestic Advisor could use hover text to explain its headings and maybe also for some additional information.

Alert for city trades added.	When a conquered city can be liberated, the raze popup shows an option for that; similar enough to a notification – in that one special case. (Since the BtS AI only trades cities when at war, notifications aren't really needed.)

Config	Can be disabled on the "Alerts" tab of the BUG menu.
See also	Implemented based on the AdvCiv4lerts class ([210](#)).

Show the alert messages in the previously unused `COLOR_CITY_BLUE`.	Some parts of the game are color-coded pretty reliably, e.g. green for research and technologies. I don't think cities are part of that color scheme; in replays, founded cities used to be shown in green. BUG used magenta (normally used for culture) for city-founded alerts.

See also	[106](#): For city-founded messages in replays, I use the player-specific text colors. [210c](#): City-founded announcements are shown in plain white.

Rationale	Colored alerts stand out more at first (which is not what I want), but, after a bit of getting used to, are easier to identify at a glance to (usually) ignore them. That's my experience with the BUG tech trade alerts anyway. `COLOR_CITY_BLUE` is surprisingly easy to read against the blue background of the Event Log but doesn't stand out much. For game text, blue tones weren't used much at all (I guess because of the blue backgrounds and water tiles). Other ideas would be `COLOR_CITY_BROWN`, `COLOR_CULTURE_RATE` or a similar new `COLOR_CITY_VIOLET` with r=50%, g=30%, b=100% – that's not too similar to the player colors of Sumer and Rome.

Show an alert when another player becomes willing to cede a city. When at war with that player, show the alert only if the war enemy would be willing to give the city in exchange for peace.

Rationale	Since the rules for city trades are fairly narrow, I'm not restricting the alert to AI players. There's no way to get a city from the AI while at war if the AI won't make peace in exchange for the city because only one side can give items in a peace deal.
	No "... will no longer cede" alert. Doesn't sound too helpful and might flicker while at war. (If it's needed after all: Don't report cities whose owner has changed, nor cities that have become the capital. Perhaps don't report anything after making peace.)

Show an alert when another player becomes willing to accept a city in trade.

Rationale	Mainly so that players know when a city becomes tradeable through the gradual spread of a foreign culture.

Show an alert when it becomes possible to liberate a city to a player other than the city's previous owner.

Rationale	Liberation to the previous owner – typically after a war – is almost always possible and rarely desirable.

AdvCiv	*K-Mod*
Show all cities that can be traded on the trade screen (and on the "Cities" tab).	Exclude "take it from our cold dead hands" cities.

Rationale	It was a good change for K-Mod (and for AdvCiv until v0.97) because the AI said that about nearly every city when at peace, and those untradeable cities were cluttering the trade screen. It may also have been part of the K-Mod changes to make unrevealed cities secret. These things are now addressed by the game rule restrictions on city trades: Can't trade for unrevealed cities and can't trade for cities without a significant amount of the new owner's tile culture.

When a human tile flips to a different owner, sleeping and fortified units of the old owner are woken up.

Rationale	So that human players can't forget about units fortified in or near a traded city. Should also be useful when border tiles flip through culture pressure; units forgotten on such tiles can even increase away-supply costs.
See also	[163](#) wakes up teleported units
Tbd.	What if a human player has units fortified on a teammate's or vassal's tile and that tile flips to a third party?

ctr (cont.)	AI for city trades	
AdvCiv		*BtS*
Except when at war, an AI civ may refuse to cede a city based on its attitude toward the recipient: • Never trade or liberate to the worst enemy		Attitude doesn't matter for city trades.

• For trade, attitude toward the recipient mustn't be below a personality-based threshold. For some leaders, the threshold is stricter (or even much stricter) when the current owner has at least 20% city tile culture. • For liberation, the current owner mustn't be Furious toward the recipient. (The AI does not refuse to *accept* a city from a disliked civ.)	

Config	The tile culture percentage that enables the stricter attitude threshold is set through `NATIVE_CITY_CULTURE_THRESH` in `Civ4GlobalDefines_advc.xml`. The personality-based thresholds are set in `Civ4LeaderHeadInfos.xml`. The respective XML tags are optional; see comments in `Civ4CivilizationsSchema.xml` about the default values.
Rationale	Rewarding good relations, especially Friendly relations (which are difficult to attain), is one of the main goals of making city trades more flexible. About liberation see under AI trade value below. The personality-based thresholds were chosen as follows: There are 10 reasonable combinations – AA, AC, AP, AF, CC, CP, CF, PP, PF and PP, where "A" means that the owner's attitude needs to be (strictly) better than Annoyed, "C" better than Cautious, "P" better than Pleased and "F" better than Friendly (which is impossible; AI response: "That would go against everything we stand for"). For variety's sake, I've tried to distribute the leaders somewhat evenly among these combinations – though some are too extreme to be justified for more than a couple of leaders. My assignment is mainly based on what notion of ethnicity a leader might have possessed. (Caveat: I'm no historian.) Consequently, it's more based on the era that they lived in than on personality. I didn't want to just make the "difficult" leaders more difficult and the "easy" ones even easier to get along with; BtS (or rather especially Vanilla Civ 4) is too one-dimensional in that regard. **AA**: Brennus, Pacal, Shaka, Montezuma – Chieftains that care more about clan than country, and rulers of city states. **CP**: Alexander, Augustus, Cyrus, Darius, Huayna Capac, Hannibal, Julius, Justinian Emperors of classical antiquity. Enlarging the inherited realm is a point of pride. Cities with little tile culture may yet be acculturated. However, in the end, it's up to the whims of the emperor. **AC**: Mansa Musa, Pericles – Pericles ruled over a city state, but there was also a sense of Hellenic identity. Mansa Musa is more of a classical emperor, meaning he doesn't easily part with land, but he also can't resist a good deal. **AP**: Ashoka, Hatshepsut, Hammurabi, Gilgamesh, Ramesses, Suryavarman, Ragnar Mostly pre-classical emperors; less interested in colonizing acquired land. Conquered cities especially are to be exploited (e.g. sold), not integrated. **CC**: Charlemagne, Isabella, Joao, Mehmed, Saladin, Suleiman, Wang Kon, Zara Y. Medieval sovereigns. Care more about feudal ties than country. They do worry that the recipient of a city might plot against them; or worse: they could be heathens. **CF**: Frederick, Louis, Willem, Elizabeth, Mao, Tokugawa The first four are rulers over early (proto-)nations. They only abandon their citizens under duress. Conquered lands are negotiable, but they're also interested in expansion. Mao: As a modern ruler also not happy to cede land, but he did cede some land to the USSR when relations weren't entirely friendly. Tokugawa: Tough one. Giving up Japanese citizens should be taboo. To minimize contact with foreign

cultures, it would seem best to trade conquered cities away. Though he doesn't really like to trade either, nor does he like foreign cities at his borders …

AF: Bismarck, De Gaulle, Napoleon, Victoria, Boudica, Sitting Bull

The first four are leaders of colonial nations, but arguably not too keen on cementing colonial rule and willing to engage in politics. And, let's say, two proud chieftains to whom the land of their forefathers is sacred.

PF: Churchill, Stalin, Peter, Catherine, Gandhi, Lincoln, Roosevelt, Washington

The first four have an aversion against giving up what they've once taken, except perhaps to a "fraternal country." The other four are modern democratic rulers that are reluctant to put a liberated people under the thumb of some less enlightened ruler.

PP: Kublai Khan, Qin Shi Huang – Want to give this combination to someone. Not sure about Qin. Kublai was practically a Chinese emperor but didn't have Chinese roots, so he shouldn't be absolutely attached to any ethnicity. On the other hand, he was an extremely powerful emperor, so he doesn't quite see why he should give anyone anything.

FF: Genghis Khan – Maybe due to his fierceness; or arrogance since his campaigns were so successful.

As for the 20% nationality threshold, I've tried 10%, same as the threshold for being able to receive a city in trade (`CITY_TRADE_CULTURE_THRESH`), but that didn't work well in situations when two war allies divvy up the spoils of a successful war. If both owner and potential recipient start at 0 nationality, then the latter would have to increase its nationality faster than the owner in order to receive the city before the stricter attitude threshold applies, which often isn't doable and also isn't consistent with the idea that having half as much nationality as the owner should suffice for a trade.

20% also has some supporting precedent in history. For example, the parts of Ukraine that Russian nationalists have been seeking to "reclaim" all have a population of at least 18% ethnic Russians (map on Wikipedia).

The AI refuses to trade away cities with a trade value greater than 4/7 of the trade value of the AI capital except when at war. If the other side is a rival, the denial reason is "you'll have to take it from our cold dead hands," otherwise "surely you ask too much." The AI refuses to accept cities with a negative trade value ("we don't want to trade this"). Even if the trade value is positive, the AI rejects small cities in very poor surroundings. Neither of these conditions apply to liberation.	The AI reponds "surely you ask too much" when asked to trade a city where its city culture is more than 50%. This only applies to teammates. Rivals are told to take any city "from our cold dead hands" except when negotiating peace. When in financial trouble, AI civs refuse to accept cities where they have 0 tile culture ("we don't want to trade this"). Cities with 0 tile culture are also rejected when they're too far away from every currently owned city. None of the AI denial checks apply to liberation.
Rationale	Want to exclude major cities in order to limit the possible gains from tech-for-land deals. One could just assign very high trade values to large cities, but excluding those cities through a denial condition should be more convenient for players; don't need to bother making offers then. I could think of various heuristics for identifying important cities (e.g. the score computed in `CvCity::getLiberationPlayer` could be a starting point), but using trade value is the easiest to implement. That's also the approach taken by UWAI for war trades – trade denial based on trade value. Cities in disorder are are problem, but the trade vaue computation needs to deal with that in any case. Performance could be a problem, but I don't think it will be. A more likely issue is a flickering city trade alert when the value of a city is near the threshold. Let's hope that this will be rare (would be a bit tedious to smooth that over). The special condition for cities on poor land (`CvPlayerAI::AI_isAwfulSite`) was

	added prior to AdvCiv 0.97 in order to discourage players from founding cities in the worst possible spots and then gifting them to the AI to improve relations. I was going to remove it in favor of a trade value check, but now I think it's safer to apply both checks.
	I've considered merely relaxing the trade value threshold when at war (instead of waiving it), but at least the UWAI code usually doesn't value peace highly enough (even when losing badly) to give up any major cities in peace deals. Saying "take it from our cold dead hands" could look bad when it's obvious to the human player that the respective city is about to fall.
	The 0-culture conditions are obsolete because the game rules now prohibit such trades. Recognizing cities that are more trouble than they're worth is, again, something that the trade value computation can handle better than some ad-hoc heuristic.
Tbd.	A downside of denial based on trade value is that no specific reason can be stated. But maybe the AI could still somehow respond with "your land is too far away" (`DENIAL_TOO_FAR`) in some situations.

The AI refuses to accept cities in trade when there are hostile land units nearby and nearby potential defenders don't clearly outnumber the potential attackers: "We are afraid of their military might." Doesn't apply when negotiating peace.	Nearby enemies (any units, not just land units) prevent liberation, but there is no such trade denial check.
Rationale	To eliminate loopholes that can make the AI look bad. Ideally, hostile transports should be counted as well and the condition should be checked in peace negotiations to make sure that the winning side isn't gaining a city that it will immediately lose to some other war party. That could be implemented, but isn't worth the effort I think.
See also	Uses code similar to the city safety check ([139](#)).
	A similar trade denial condition described among the [rule changes](#) deals with units hostile to the *current* city owner.

The AI refuses to accept cities that are under occupation if the resistance is coming from a third party: "Maybe we'll change our minds in a few years."	Occupation and disorder don't affect trade denial.
Rationale	Computing trade value during disorder isn't a problem, but don't want humans to deliberately let a revolt happen before trading a city to the AI. It's tempting to prohibit all city trades during disorder, but it would seem strange for the recipient to insist that a revolt in its own favor be suppressed before accepting the city. One could argue that the city might flip on its own, but it's better to address that angle through the trade value computation (certainly the AI should accept a city as a gift even if it will flip).

Capitulated vassals refuse to trade cities to their master unless the capitulated vassal is above the population threshold for breaking free and will still be above the threshold after losing the city: "Surely you ask too much." AI civs don't trade cities to their capitulated vassals if that would put the vassal above the threshold for breaking free or if the vassal is already above the threshold: "You've grown too powerful for us."	Vassals can't cede cities to anyone and can't accept cities from their master.
See also	[130v](#) about the mod's general approach to capitulated and voluntary vassals.
	See the [rule changes](#) to city trades about restrictions that apply to all vassals.

Rationale	Capitulated vassals shouldn't ruin their chances of breaking free. Master civs shouldn't help their capitulated vassals break free.

When close to a Conquest victory, the AI refuses to trade away any cities except when negotiating peace.

When close to a Space victory, the AI refuses to trade away high-production cities and cities that are producing a spaceship part.

Stated reason in both cases: "We'd rather win the game"

Rationale	The Space conditions will rarely matter, I expect, because the trade value conditions already prevent the AI from ceding major cities. For Culture victory, that should 100% be the case.

An AI civ with a war utility value above 30 may refuse to accept a city in a non-liberation trade when the city is insignificant compared to the economic output of the recipient and the war utility value. (If UWAI is disabled, a coarser heuristic based on attitude and military power is used instead of war utility.) Stated reason: "We have our reasons"

Rationale	To avoid signing a peace treaty. On the one hand, gaining one decent city is usually more than the AI will accomplish by actually declaring war on a human player; on the other hand, when humans can rely on getting a peace treaty in exchange for a city, they may exploit that by recklessly exposing themselves or by buying time for a peaceful victory in the endgame.
	"We have our reasons" will alert humans about the high wa rutility value; probably obvious anyway in the rare cases when this will come up.
Tbd.	Should perhaps still add a special message to the city trade alert – to make sure that players don't feel encouraged to keep an eye on the City Trades tab for AI war plans.

By default, 0 trade value is assigned to liberated cities, meaning also that the AI will respond that no trade is possible when asked what it will give in exchange for liberation.	At peacetime, the game rules and UI prevent cities from being traded (except between two human players).
When negotiating terms for peace, liberation has no impact on a city's trade value.	Liberation never affects a city's trade value.

Rationale	Mainly because I don't want to render the liberation choices in the conquest and colony (Alt+F1) popups useless. If the AI pays for liberation, even if it doesn't pay much, it'll always be preferable to liberate cities through trade. Therefore, I'm interpreting liberation as the recipient having a rightful claim to the city and insisting on getting it for free (granting peace isn't really a payment). Admittedly not very realistic. In gameplay terms, the reward for liberating a city is the relations boost (see a few boxes below).
	It would be easy enough (by reverting a one-line change in `CvPlayerAI:: AI_counterPropose` marked with `advc.ctr`) to let the AI make a proposal that leaves the left side of the trade table empty when asked what they'd give for the liberation of a city, but stating that no trade is possible (as in BtS) seems less confusing to me. (Although a trade might actually be possible, namely, when the AI has a city that it can liberate in return.)
Tbd.	`CvCity::getLiberationPlayer` may have to be simplified because, when deciding whether to trade a city away, players may want to take into account whether the city will soon become eligible for liberation – meaning that the the recipient won't pay for it anymore. That's currently impossible to gauge.

When considering to liberate a city in response to a human help or tribute request, the AI decides	Human can request liberation, but the AI will decide based on the regular trade value of the

based on a reduced trade value that only takes into account how valuable the city is to the AI civ.	city.
Rationale	If humans are expected to liberate cities for free, there should be a real possibility of the AI doing the same thing. The AI will also liberate cities without human importuning; see below.
When considering a trade in which each side liberates at least one city, a reduced trade value is counted for all liberations. That trade value accounts only for the value that the city has to its current owner; although, if the current owner is human, then that value is capped at 50% of the value that the city has for the AI recipient.	Regardless of liberation, the AI can never trade cities for other cities.
Rationale	To facilitate reciprocal liberation, which is a win-win.
When a human player asks the AI to liberate a city and offers trade items in exchange or asks the AI for a price, the AI computes the city's full trade value, i.e. as if the liberation conditions didn't apply.	The UI doesn't allow humans to propose city trades and the AI will say that no such trade is possible.
Rationale	Liberation shouldnt ever be a handicap for the liberation player. Unlike AI players, human players aren't going to "feel too entitled to pay" for a city, so they should be allowed to pay regardless of liberation.
For non-liberation trades, the AI evaluation is broken down into two similar components: One value expressing how much the current owner want to keep the city and one value expressing how much the recipient wants to acquire the city. When evaluating a trade proposal, both values are added up and, if the AI civ is the recipient, capped at two times the value of acquiring the city.	K-Mod comment: "*The way this function is currently used is that it actually represents how much the current owner values* not *giving the city to this player. For example, if this player currently controls most of the city's culture, the value should be* lower *rather than higher, so that the current owner is more likely to give up the city. Ideally the value of receiving the city and the cost of giving the city away would be separate things; but that's currently not how trades are made.*"
Rationale	The BtS approach is workable for tech, which mostly benefits one side without hurting the other – so the trade value is essentially equal to the tech cost –, and for gold, which is mostly zero-sum – the trade value is two times the amount of gold –, but ill-suited for city trades, which can be zero-sum, but can also involve cities that would be much more valuable to the recipient than to the current owner.

The cap based on the recipient's value is supposed to ensure that the AI doesn't pay a high price for a city that is more valuable to its current owner. |
See also	Afforess has added a function `AI_ourCityValue` to Rise of Mankind – which is mostly a copy of `AI_cityTradeVal`.
Tbd.	Ideally, all trade evaluation functions (perhaps also the ones checking denial conditions) should be broken down into the perspective of the giving and the receiving side. E.g. how much the recipient civ benefits from a traded tech versus how much he civ imparting the tech would like to withhold it from the recipient. This structure could be imposed through a simple object-oriented framework.
AdvCiv	*BtS/K-Mod*
The trade value computation is further broken down into an economic "asset" value, an amortization modifier and modifiers for strategic	A simple, yet arcane, formula based on population (K-Mod: also highest-ever population), city culture, city tile culture, game turn (K-Mod:

considerations. The asset value doesn't just cover the current economic value, but also tries to (crudely) estimate its economic potential (possibly hampered by culture pressure). Estimated expenses are subtracted.	also turn founded and turn acquired), prior ownership of the recipient and the trade values of all bonus resources in the full city radius (K-Mod: lower weight for the outer ring). K-Mod added an evaluation great wonders and Holy City status (but national wonders of the current owner and mundane buildings still weren't covered).
The amortization modifier accounts for speed and other game settings.	No meaningful game speed adjustment.
The strategic modifiers account for the (old or new) owner's total city count, number of cities on the same landmass, whether the city might flip, whether the extra population might soon let the new owner win (Domination or Diplo victory), the tactical situation (city imminently threatened or already being evacuated) and relations – in particular whether the either side could easily conquer the city instead of making a trade or (reconquer it) after making the trade.	None of these are directly addressed.

Credits	I've reviewed Afforess' code in Rise of Mankind (RoM; SVN revision) but only adopted (in part) code dealing with religions and corporations present in the city. The RoM code takes into account production costs of buildings, but buildings aren't public knowledge in AdvCiv. Therefore, and to save time, my code uses the building count for a vaguer estimate of building utility. There are checks for imminent danger of conquest, same landmass as capital and financial trouble; the new AdvCiv code handles these things in a slightly more sophisticated manner. This CFC post suggests that the RoM AI often overvalues cities.
See also	104d: The asset value is also used by UWAI. The tactical evaluation when at war is based on 139. 045 makes most city buildings secret.
Rationale	The K-Mod formula doesn't seem adequate now that city trades will be more commonplace. (Apparently, the K-Mod hadn't been tested much either; there was a major bug in the computation of a city culture multiplier. That code is now deleted; the final version is from 22 Mar 2020, after this Git commit.) I don't think the exact numbers matter much, but some of my additions could be important for preventing the AI from overpaying for human cities.

The AI city evaluation is a bit stacked against human city owners through a distrust multiplier (between 2/3 and 5/6 depending on attitude) and AI refusal to compensate humans for avoidable hardships (e.g. losing a national wonder).	No anti-human bias in AI city evaluation.

Rationale	Really just humility on the part of the AI: The human player probably knows better how valuable his or her cities are and wouldn't give cities away that are very valuable. Hopefully, no bias will be apparent.

When trading with a human, the AI trade value computation ends with a multiplication by a secret factor near 1 that gets re-randomized unpredictably (hopefully) every few turns. Then	The final result is rounded to a multiple of 10 and the computation doesn't use any sensitive information.

the usual rounding to a multiple of 5 is applied	
Rationale	To obscure the specifics of the computation. In particular so that the AI prediction of a (near-)future war and minor AI information cheats (when evaluating human cities) aren't exposed.
See also	[136b](#) changes the standard rounding of trade values from a multiple of 10 to a multiple of 5.

The AI recognizes awful city sites, doesn't accept such cities in trade and razes them after conquest.	If a city is close enough and has enough culture, the AI accepts it, and doesn't normally raze it.
Rationale	The BtS behavior can be abused by gifting the AI useless cities, and the AI will even be thankful for it (fair trade diplo bonus). See strategy advise [here](#) (CFC). That strategy still works, the city site just mustn't be *extremely* bad.

AdvCiv		*K-Mod*
When "making this deal work" or determining "the price for peace" (`AI_counterPropose`), the AI can add more than one city to the trade table, but only if a single city and all available non-city trade items aren't enough to cover the current gap in trade values. (This doesn't apply to peace deals between AI civs. There's a separate logic for those, which still allows only one city to change hands.)		K-Mod comment: "*We're only going to allow one city on the list. (For flavour reasons.)*" And only when at war. Looks like BtS had always put a city on the table when at war; K-Mod only adds the city to a list of trade items and later greedily picks items from that list to find the best fit for the gap in trade values (that's still the basic approach in AdvCiv).
Rationale	Seemed like the easiest way to allow multiple cities without changing the basic algorithm. Don't want the AI to propose a city trade over a tech trade. Trading multiple cities for peace rarely ever makes sense. I want to allow it for humans just so that the AI doesn't wrongly claim that peace isn't possible. At peacetime, in the late game or on large maps, trading one city for two or two small cities for a declaration of war may not even be so unusual.	
See also	[104h](#): Slightly changes the conditions under which the AI adds cities to its initial proposal for a peace deal.	

`AI_counterPropose` prioritizes cities for trade based on the difference between the owner's trade value and the recipient's trade value, i.e. it tries to find positive-sum (as opposed to zero-sum) trades.	The single city that the AI will propose is chosen based on `AI_targetCityVal`.
Rationale	Seems like a better heuristic. More importantly, using `AI_targetCityVal` leaks information about where the AI will focus its attacks. Not a well-known exploit, but, in [this](#) CFC post, one player is onto it.

`AI_counterPropose` doesn't add city liberations to the trade table except (see a few boxes above) when the trade value computation ignores liberation.	
Rationale	This only applies in the rare situation when a human and AI civ both have at least one city that they can liberate. It's difficult for the AI to decide which cities the human player might want to liberate or accept – at least in some situations, and it's in any case a bit tedious to implement. Letting the human player put the cities on the table shouldn't be too much of an inconvenience; the AI will then fill in other trade items as needed.

AI diplo memory about liberation is increased or reduced by 50% depending on the (non-liberation) trade value of the liberated city.	

See also	In line with similar changes to raze and nuke memory: [130q](#)
When computing the effect of a liberated city on trade memory ("fair and forthright", "traded with our worst enemies"), the trade values of both sides are counted (not just the value of the old owner as in some other cases -- see higher up in the boxes about liberation trade value), but the total is reduced a bit.	To my knowledge, the full trade value is counted by BtS and K-Mod.
Rationale	The AI shouldn't be too grateful or upset when a city is returned to its rightful owner (but +2 from liberation alone would seem a bit too meager). Counting only the loss of the old owner wouldn't make sense for the enemy trade value. Actually, it would make sense to count only the *new* owner's trade value (that's the player whom third parties resent), but that would be awkward to implement as BtS normally uses just one trade value that accounts for both sides.
When asking for help, the AI may ask for a city if that city has significantly higher value for the AI civ than for the human civ, and if its trade value isn't much higher than that of the tech that the AI would otherwise ask for or if there is no tech that the AI could ask for.	The only asks for help in the form of free tech.
Rationale	No important reason. To incentivize liberating AI cities, for more variety in help requests and to enable help requests when tech trading is disabled. I feel that the AI shouldn't ask for help in the form of zero-sum trades, e.g. gold per turn. Cities can help the recipient much more than their current owner.
When demanding tribute, the AI may demand a city if that city isn't far more valuable to its current owner.	Tribute can take the form of tech, world map, a resource or gold (lump sum).
Rationale	Flavor, variety, give players that are behind in technology some option to avert war. The demanded item being too valuable is less of a concern than with help requests.
See also	[104m](#) allows the AI to demand gold per turn and lets the AI consider the various types of tribute demands in a randomized order until a valid demand is found.
For each city that can be ceded, its AI owner checks every 20 turns, the first time being based on the city's id and thus unpredictable, whether that city can liberated or is significantly more valuable to another civ. If so, the owner may contact that other civ, offering the city.	Same rhythm (though, in principle, entirely predictable). Due to the trade denial conditions, only liberation is considered. Liberation to the previous owner is ruled out. So, typically, it'll be liberation to the original owner of a city.
Rationale	I'm extending the contact-to-liberate mechanism to cover also AI offers for city trades to both human and other AI civs.
While a city is evacuating through change [139](#), the AI checks on every turn for a third party that the current owner might want to cede the city to through liberation.	
Rationale	What a human player would also do. Non-liberation trades of a threatened city to a third party are disallowed by the (new) city trade [rules](#).
When offering a city for liberation, the AI asks the recipient to also liberate a city if possible. Usually, it's not possible and then the AI offers to liberate the city for free.	If the AI offers a city, it's always for free.
Rationale	Consistent with how the AI treats human offers for liberation (see somewhere higher up).

The AI does not offer (free) liberation when the city is of considerable value to its current owner, if it's almost as valuable to the current owner as to the liberation player or if it's too valuable to the new owner. The value threshold for that last condition is based on attitude. All the thresholds scale with the game progress and speed. The AI also tries to avoid empowering (non-colonial) vassals too much.	For offering liberation, the AI owner needs to have at least Cautious attitude toward the liberation player.

Rationale	Don't want the AI to hurt itself severely through a city gift, and don't want it to boost a rival too much. As for vassals, liberating a city that'll will be much more useful to the vassal is a good move, but the vassal mustn't become so strong that it'll be able and willing to break free.

The AI refuses to cede a city to its previous owner (trade denial: "Maybe we'll change our mind in a few years") for, on average, about 15 turns.	The AI never contacts the previous owner for liberation. It could, in theory liberate to a previous human owner who asks for the city as a gift or tribute.
The AI doesn't offer to liberate a city for free when it thinks that war with the liberation player is a possibility in the medium term.	No such strategic considerations.

Rationale	I'd like the AI to be able to return a conquered city to its previous owner, even without payment, when the two civs are no longer at each other's throat and the city isn't doing much for the new owner. The AI being too quick to return a city will look very bad, so perhaps these conditions will have to be tightened further. Conquered AI cities permanently enclosed by foreign culture also look bad though.

After conquering a city, the AI may immediately liberate it if the liberation player is a third party. The same conditions as described above for free liberation apply except that no trade denial checks are performed.	Only human conquerors can immediately liberate a city after conquest.

See also	Among the rule changes to city trades, the (unchanged) BtS special rules for liberation after conquest are described. Those rules also apply when the AI considers liberation upon conquest.

Rationale	Easy enough to do; as a rule, human and AI should play by the same rules.

The AI may offer a city for trade if it is much more valuable to the recipient or if the city isn't much more valuable to its current owner and the recipient has far more assets (population, land, tech, buildings) than the current owner. The AI then tries to find a city of the recipient that satisfies the same conditions – to propose a one-for-one city trade with additional trade items for the side that gets the less valuable city. Usually, there is no such city, and then the AI looks for arbitrary trade items (i.e. tech, gold) that the recipient can offer in return. AI city trade offers to human players include a discount.	

Rationale	See the rationales given at the start of the city trade documentation. AI-to-human trades aren't crucial, but make the AI appear a bit more alive.

See also	The discount is the usual incentive for humans to actually consider the AI offer. Through 026, the AI may also offer more gold than it normally would.

When two AI civs are negotiating a joint war, the civ that is already at war may offer a city (in principle several, but I doubt that will happen) in	The civ already at war may offer one or two techs and/or gold.

exchange for the declaration of war. Unlike in other city trades, the offered city doesn't have to be more valuable to the recipient than to the current owner, at least not when the war is going badly.	
Rationale	Not sure how frequently this will come about, but it could be pretty awesome when an AI civ that seemed already doomed brings in a powerful neighbor as a war ally by ceding a city.
Tbd.	Such trades would work as a matter of course due to my changes to `CvPlayerAI::AI_counterPropose` – but that function isn't actually used for war trades; perhaps it should be (see comments in `CvPlayerAI::AI_proposeWarTrade`).

123	Close loopholes
See also	Meatgrinder: [139](). Worker stealing ([010]()) is a bit of a loophole too I guess. Flat maps for lower distance maintenance: [140](). Gifting GP for diplo victory: [141](). Neutral units shielding cities from nukes: [kekm.7](). Gifting the AI useless cities: [ctr](). Gifting nukes: [143b](). Extra gold in Advanced Start with Expansive trait: [kekm.11](). Rival wonders under construction deduced through Sabotage Production mission cost: [120d](). Bumped units have all their movement points spent: [163](). Surrounding a city with friendly units doesn't stop Barbarians from trying to conquer that city: [083](). Can't pillage own resources to stack up overflow production (and generally make it harder to generate overflow): [064d](). AI peace proposal giving away current target city: [ctr(AI)](). No extra Golden Age turn when completing the Taj Mahal during a Golden Age: [001x](). Can't circumvent gold-per-turn limits through subsidies: [133](). Two nukes don't reliably kill all city defenders: [650](). Tile yields don't give away unrevealed resources on foreign tiles: [182]().

Loopholes closed by K-Mod (not a complete list I think):
AI never agrees to Cease Fire. Spread Culture spy mission doesn't increase city culture. |
| *Tbd.* | If I ever want to write AI code for Cease Fire: `CvPlayerAI::AI_considerOffer` would be the place (see comment near the start of that function). |
| **123a** | Can't gift Missionaries to bypass Theocracy |

AdvCiv	*BtS*
Can't gift a Missionary to a civ in Theocracy unless the Missionary matches their state religion.	Can bypass the Theocracy restriction by gifting Missionaries. The AI will normally use them too.
Can't gift Caravels to a civ without an OB agreement.	Can gift Missionaries inside Caravels in order to spread a religion without OB.
Rationale	The Theocracy restriction is pretty pointless this way. And keeping borders closed should keep Missionaries out, period.
Credits	Kek-Mod fixed the Theocracy part independently. I hadn't thought of cargo units (other than Caravels), so I've adopted part of the Kek-Mod fix.
See also	[kekm.4]()
123b	*Unused*
123c	Chains of cargo units

Land units can only be loaded as cargo if they	Loading and unloading don't require moves

have at least 1 move left. No change to air units; can still be loaded right after rebasing.	(although Loading consumes all moves). Can move land units across arbitrary distances over water within a single turn by using a chain of dispersed cargo units.
123d	Reduce war success when trading away a city while at war
When a city is traded away, and the old owner is at war with any earlier owners of the traded city, the war success of the old owner against these earlier owners is reduced by 25.	A player can conquer a city (+25 war success against the enemy), gift it to a war ally (no change to war success), wait for the enemy to conquer the city again (+25 war success against the player's ally), and repeat. The war enemy will then assume that the war against the player is going very badly. [Actually, BtS counts only 10 war success points per city; BBAI has increased that to 25. So this isn't as much of an issue in BtS.]
Rationale	Not so easy to exploit, but occasionally leads to quick capitulation. Reducing the war success against *all* earlier owners can be overkill, but this is difficult to avoid.
Tbd.	The war success counted for a conquered (or traded) city should be based on the importance of the city. Currently, every city counts as `WAR_SUCCESS_CITY_CAPTURING` – which should remain the base value, to be multiplied by some modifier – except the capital, which already has a modifier of 1.5 (was 2 in BBAI; I've reduced it). The function that records war success is `CvUnit::setXY`.
123e	Privateers can no longer plunder gold from Barbarian cities
Credits	Civ 4 Reimagined 1.2
See also	033 prevents Privateers from plundering vassal cities.
123f	Fail gold only for great wonders and at most once per wonder
Tbd.	Perhaps convert only 50% of the invested production into gold and the rest into city culture. That should be a separate change id though because it's a balance change rather than just closing a loophole. For now, fail gold remains a valid tactic, and this gives wonders with very weak abilities (e.g. Chichen Itza) at least *some* use.
AdvCiv	*BtS*
When a great wonder is completed in a city, that wonder is immediately removed from the production queues of all other cities, generating fail gold.	After a message about a rival finishing a wonder, another turn passes before the game removes the wonder from production queues and generates fail gold.
Exception: Members of the team that has completed the wonder never receive fail gold.	No such exception.
When a player has invested production into a wonder but removed that wonder from the production queue, no fail gold is generated.	Fail gold is generating regardless of whether the wonder is queued. This allows crafty players to cash in multiple times per wonder.
World projects are treated just like wonders. Units and national wonders never produce fail gold.	Units, national wonders, great wonders, world and team projects are all treated alike, meaning that fail gold can also be obtained from e.g. Missionaries (see this CFC thread).
Rationale	The Wealth process should be used for converting production into gold, and that's already a pretty efficient conversion. Award fail gold only when it has to be done,

	namely when a civ is beaten to a wonder by a rival. If there was no such compensation, wonders would become too unattractive and frustrating. (Civ 6 reportedly has that problem.)

AdvCiv	*K-Mod*
Units never produce overflow gold.	K-Mod uses excess production on additional units of the same type. Overflow can still occur I think, but only on units with a national limit. As for buildings (no change): BtS 3.19 with the unofficial patch converts production into gold, but cancels out building-specific modifiers like Stone for Walls.

Rationale	A side-effect of disabling fail gold for units, and I don't think that overflow gold should be a worthwhile tactic.
Config	`MAXED_UNIT_GOLD_PERCENT` in `GlobalDefines_advc.xml`.
Tbd.	Once I remove the whip ability (hurry through population), it should be all but impossible to use the overflow rules for converting production into gold.

123g	Need to choose research a.s.a.p. at the start of a game
Once the player has founded a city, a choose-research popup appears. If the player ends the turn without addressing the popup or sets research back to "no technology" after dealing with the popup, the AI chooses a tech for the player and the popup appears again on the next turn. After founding the first city, the choose-research popup appears before the choose-production popup.	A choose-research popup appears one turn after founding the first city. If the player manages to avoid choosing a tech to research, then, during the first five turns, research is stored as overflow research; that research is spent once the player does select a tech and research modifiers are applied based on the selected tech. After five turns, the AI chooses for the player. Not choosing a tech for five turns can yield a little bit of extra research due to known-tech modifiers; see the description [here](#) on CFC. The choose-production popup appears first.

Rationale	My guess is that the Civ 4 developers wanted to delay the choose-tech popup by one turn to make the first turn easier for beginners, and set the limit to five turns because the player might move the Settler around a bit before founding a city. For this mod, this wouldn't make sense, and is a needless (and exploitable) complication. Choosing tech before production should actually be a bit easier for experienced players than vice versa.
See also	[Thread](#) with detailed info on the subject (and a post by me)

124	Restrictions on trade
AdvCiv	*BtS*
Trade connections only along revealed plots, i.e. in order to establish a trade route, the owner of the first city needs to explore the second city and the roads/rivers/coast leading there.	Plots owned by another civ don't need to be revealed. I.e. by revealing a path of road, river and coastal tiles up to any foreign tile connected to the foreign capital, trade routes are established with all foreign cities connected (along owned tiles) to the foreign capital, including unrevealed cities.

Resource trades require a connection to any foreign city that is connected to the capital; the capital itself doesn't have to be revealed. I.e. pretty much no change. No trade routes with cities in occupation, and no trade routes with cities whose owners are in anarchy.	Resource trades require a connection to the foreign capital.

Rationale	The BtS concept isn't that unrealistic. If one side knows half of the way and the other side the other half, trade can be conducted in the middle. Soren Johnson explained this in a Twitch video, naming trade between Han China and Rome as example.
	I think it's better for gameplay when the other city has to be revealed because this rewards repeated exploration of rival territory, and the game tends to be more fun when you keep updated on rival activities. Also makes Scouts more useful. Trade between Han and Romans is better modeled as resource trade.
	The BtS rules are also inconsistent with K-Mod's treatment of unrevealed cities as secret; the trade routes listed on the BtS city screen give the cities away.
	The occupation/ anarchy rule is just for added plausibility. I've considered excluding blockaded and plundered cities as well, but such cities could still trade across land, i.e. unless the whole landmass is blocked, in which case trade routes are already severed.
	Should tiles owned by a rival block trade unless there is an OB agreement? I think the BtS rule (trade with third parties only blocked when at war) plays better and is arguably also more realistic. E.g. the Strait of Hormuz remained open despite the 1995 sanctions against Iran.
Tbd.	Would like to change the way that cities are matched. The current algorithm creates long-distance trade routes between large cities, but these cities aren't usually the borderland trade hubs that would realistically profit from trade passing through. Boils down to the question if trade routes should be represented from end to end or hop by hop.

For resource trades, only one side needs to be able to reach the other.	Trade connections are always symmetrical, so this isn't an issue.

Rationale	Want a civ that establishes a trade connection to be able to trade resources right away (as in BtS), instead of having to wait for the other civ to e.g. also research Sailing.

Trade along rivers doesn't require any tech; back to how it worked prior to BtS. Trade along unowned coasts still requires Sailing. Help text says "Enables trade on Coasts outside own borders".	Trade along *owned* rivers works from the beginning, along unowned rivers only with Sailing. Sailing help text just says "enables trade on Rivers"/ "...Coasts".

Rationale	Not sure why this change was made in BtS. The distinction between owned and unowned rivers is pedantic, and confuses players (people ask about this now and then on CFC). Rivers now work just like roads when it comes to trade.
Tbd.	Want rivers to speed up movement as in Alpha Centauri. Fishing could enable this.

An AI civ is willing to sign Open Borders with another civ only once it has revealed a land tile owned by that civ, or if AI attitude is one level above the normal OB threshold, i.e. at Pleased in most cases. Otherwise: "We would have nothing to gain." Exception (added in AdvCiv 0.99): If all revealed	OB based only on a personality-based attitude threshold.

tiles are located on continents where the AI civ doesn't have any cities, then the tile will only cause the AI to open its borders if a trade connection exists between the AI capital and the tile (through Sailing or Astronomy). Relevant mostly for mod-mods: If the AI has a unit adjacent to a tile that the unit can't currently enter and will be able to enter through OB, then the AI is willing to sign OB based on the normal attitude threshold. (Even if the tile is not a land tile and doesn't have a trade connection. I don't think this can easily happen with the BtS/AdvCiv roster of units.)	
Will sign OB at attitude one level below the normal OB threshold when sharing a war but, again, only if there is a revealed owned tile.	AI signs OB regardless of attitude when sharing a war.
The AI initiates OB with another AI (or proposes OB to a human) 10 turns (on average) after becoming willing to agree to an OB proposal. The delay in between OB proposals is at least 40 turns (45 on average).	20 turns after becoming willing, at least 20 turns delay in between proposals (30 on average).

Rationale	Don't want the AI to sign OB when it clearly won't benefit from the trade – at least not at Cautious attitude. Once the AI is willing to sign OB with another AI, it shouldn't wait for another (up to) 20 turns.
Credits	Got the idea for signing OB more quickly from Civ 4 Reimagined 1.3. They even set it to 5 turns.
See also	130i reduces the diplo bonus from OB that don't actually benefit the AI. This makes OB less with remote AI civs less of a matter of course and thus makes it more important to stop the AI from proposing OB over and over (through an increased contact delay). 130r: adjusts contact delays to the game speed setting.
Tbd.	The condition for being able to immediately enter foreign borders with a unit is at best a kludge. Let's assume that some mod-mod has a Medieval Cog unit that can reach foreign lands across Ocean but (unlike Caravel) can't enter foreign borders without OB, and assume that Astronomy is still required for cross-ocean trade. Don't want the AI to be unwilling to sign OB when a Cog reaches another continent. The current AI code will only make the AI willing to sign OB when the Cog is right at the borders; as soon as it turns around, the offer will be off. I don't think it makes much sense to allow cross-ocean movement without allowing cross-ocean trade, so maybe I should just remove the kludge. It's also potentially expensive in terms of computing time.

The AI makes it a high priority to build at least one naval explorer once it knows Sailing.	The AI prioritizes exploration only until meeting all civs.
AI considers Galley to be suitable for exploration.	Unit AI types for Galley are only Transport and Attack. Before Caravels, only Work Boat and Trireme have the Explore AI type. (Though `CvPlayerAI::AI_unitValue` may have considered Galley as an exploration unit regardless of the types listed in XML.)
Rationale	Due to the trade route rule change, the AI needs to explore coasts more reliably. Should use Galleys too because Work Boats get picked off by Barbarians at some

	point and Triremes come too late.
See also	905a increases the speed of Galleys; this makes them more suitable as explorers.

Trade can pass through hostile plots if these plots are affected by a naval blockade by a party that is not hostile to the trading civ. In particular, the trading civ itself could blockade the plots. The AI does not use blockades to enable trade. ~~Water tiles with a friendly unit can be worked despite being blockaded.~~ Disabled again. It's a bit more sensible but too unimportant to bother.	Trade can never pass through hostile plots and blockades can only prohibit trade. Blockade prevents tiles from being worked despite the presence of units protecting the tile. Those units will have to engage the blockading unit to make the tile workable.
Rationale	Occasionally, a declaration of war severs important maritime trade connections between cities of one of the war parties or with some third party. This should not happen to a civ that has naval superiority. Would be too much work to get the AI to use blockades this way. Should the mere presence of a unit in a (land) tile cause that tile to be blockaded? How do you operate, say, a winery in a tile that is occupied by enemy military? Could argue that wine remains available from a stockpile for one turn, and after that turn the enemy has had an opportunity to pillage the tile. Would be nice to have the option of blocking a resource for several turns without pillaging, but I worry that resources would get disrupted too frequently for just one turn, causing (human) trades to be canceled and citizens to be reassigned. Anyway, `CvPlot::isTradeNetwork` would be the place to implement such a change.

A structure (route, fort or city) can only be part of a civ's trade network, if that structure is revealed to the civ.	If the tile is owned by another civ, then not even the tile needs to be revealed in order to carry trade. Otherwise, it's sufficient if the tile is revealed, even if the tile is fogged and no structure existed the last time that it was visible. (Structures on unowned tiles are unusual though, so this isn't really a problem in BtS.)
Rationale	Consistent with the (AdvCiv) rule that trade only works along revealed tiles and can only be conducted with revealed cities.
See also	Somewhat important for 004z, which lets the trade layer show trade connections on unowned tiles. Don't want the Trade layer to give away structures hidden in the Fog of War. 001i fixes other issues with routes in the fog of war.

Tiles workable by a city on a different landmass can be connected to adjacent water tiles through a route. This means, a resource on an island within the workable radius of a (mainland) city doesn't require a fort in order to be connected.	Whether a tile is workable doesn't matter for trade connections. A connection between land and water requires a river, city or fort. This means, an offshore resource that is worked by a mainland city can (unless there is a river on the island) either be worked for its maximal yield (no fort) or be connected to the trade network (fort) – but not both.
Rationale	This keeps confusing players, see e.g. this thread on CFC. My change removes the awkward choice between connecting the resource and exploiting its improvement yields. It doesn't remove the confusion in situations when a tile is owned by the player but not workable. If any owned route would connect to coast ... – see the notes below regarding the rule change from AdvCiv 0.95 to 0.96 about that.

	The different-landmass clause wasn't added until v0.96. At first (v0.95), I thought I wouldn't need to make this a special rule for islands. However, allowing all workable tiles to be connected to water through a route meant that no coastal city/ fort was required for a long-distance coastal trade connection in the early game and that a single route sufficed for connecting any coastal resource. In v0.96, there was the additional requirement of owning no cities on the resource's landmass. That seems like a needless complication to me now.
See also	040: The AI transports Workers to workable tiles on other landmasses.
	Other mods have addressed this problem as well, Realism Invictus at least, though I don't remember how exactly.

Naval trade passing through a fort doesn't require a route on the fort.	Trade can pass through land tiles only through a route, river or city (has a route built in).
Rationale	Looks like an oversight that happened when forts were given their additional abilities by the BtS expansion.
Credits	Bug report (toward the end of the post) with screenshot

Mercantilism allows trade routes with vassals of the same master (and with the master). Help text (Civics screen, new-civic popup, Civilopedia) says that Mercantilism blocks only trade routes with rivals.	Only with the master. (And the master can have trade routes with its vassals – no change.) Help text says that trade routes with all foreign civs are blocked. I don't think the exception for vassal/ master (presumably introduced by Warlords) is mentioned anywhere.
Rationale	This minor rule change makes the improved help text easier to phrase.

125	Culture from trade routes disabled
AdvCiv	*K-Mod*
Trade routes generate only raw commerce. ~~Trade routes only generate culture if the city tile already has some culture (more than 0 points) of the trade partner.~~ (Decided to disable it entirely instead.)	Trade routes generate foreign tile culture in addition to raw commerce.
Rationale	The only real upside of trade culture is flavor, and I don't think this justifies the added complexity. The confusing part is that trade culture could also be added to city culture (but it isn't). Tile culture from faraway trade partners may be a disadvantage in local border disputes; difficult to say.
	Unimportant culture in plot help text is another problem. Very small percentages aren't shown, but this means plot culture sums up to just, say, 97% sometimes, which is confusing. (99% is OK – everyone's used to that.)
	Another issue: The Nationality bar on the city screen can show only culture of up to four different civs, and this can't be changed within the SDK.
Config	`USE_KMOD_TRADE_CULTURE` in `GlobalDefines_advc`

126	Changes to the later-era start option
See also	[138](#) changes the religion assignment when starting in a later era [307](#) makes Machine Gun ineligible as a free starting unit (because it can't stop Barbarians from pillaging). [314](#) allows tribal villages when starting in the Medieval era.

AdvCiv	BtS
Added one free tech for each era after Ancient (except Future) to each difficulty setting that grants free technology to the AI. These free techs are only granted when starting in the respective era. Similarly, the human player receives additional free techs on Chieftain and Settler. On Settler to Chieftain and Monarch to Immortal, the new free techs are Mathematics, Machinery, Printing Press, Steampower and Plastics. In addition to those, AI Deity and human Settler receive Ironworking (only AI), Alphabet (only human), Feudalism, Gunpowder, Steel and Refrigeration.	When starting in a later era, every civ receives at least all Ancient techs. Neither human nor AI receive free techs beyond those.
On difficulty settings that grant free units to the AI, more units are granted when starting in a later era: another defensive unit for each era beyond Ancient, another Worker for every 2 eras and another Scout or Explorer for every 3 eras.	Every civ receives additional free units in later eras. AI civs also receive free units granted by the difficulty setting. All free units are chosen based on the start era, e.g. Riflemen as defensive units when starting in the Industrial era.

Rationale	Games starting in later eras tend to be much easier to win than normal games, surely in no small part, because the AI doesn't get a proper head start. No tech advantage, and the extra free units are relatively few compared with the total number of free units, which may even allow a human player to conquer a high-level AI right away.
Config	The free tech is configured in `Civ4HandicapInfo.xml`.
See also	[301](#) prevents Barbarians from using Copper and Horse until these resources are connected by some city. This used to be a problem with Classical starts – Barbarians would have Axemen and Horse Archers right away, whereas the (AI) civs needed some 50 turns to establish access to Copper or Horse.
Tbd.	The free techs aren't carefully chosen. At some point, I want to change the era of some techs; can reconsider the freebies after that. Later-era starts probably still have balance issues that make them too easy or too difficult to win; I've never even finished a test game starting in a later era. Would be nice to make at least Classical and Medieval starts playable; these aren't *that* different from normal games, but could be different enough to be refreshing.

A tech can only fulfill a requirement of another tech when all tech required by the first tech are known.	For example, in the Earth1000AD scenario, India starts with Paper but doesn't have any of its prerequisites. This does not prevent India from researching Education (enabled by Paper).
Rationale	I'm not sure why I've placed this change under this change id. It seems that I didn't document it at the time. Could be relevant for free techs granted in later eras but isn't currently relevant. Also not relevant for Archery as a free tech on Monarch difficulty in

the Ancient era because Archery is a dead-end. Used to be relevant for free Pottery on "King" difficulty ([250a](#)), but that difficulty setting no longer exists.	
The point and XP thresholds for Great People (GP) are not affected by the start era.	When starting in Renaissance, the GP thresholds are decreased by 10%, by 20% for Industrial, 30% Modern and 40% Future.
Rationale	The initial GP threshold is much lower when starting in Renaissance (namely 100) than when a normal game reaches that era, and, with the two initial cities with two population each, free Forges and Mercantilism, civs can and will immediately breed Engineers. This is pretty strange, so I was going to increase the initial GP thresholds for Renaissance and later starts. However, when starting in Renaissance, it takes way too long to discover essential Renaissance techs like Astronomy, and easy access to Great People can mitigate this problem. Now I'm simply using the standard thresholds as a compromise.
Config	`Civ4EraInfos.xml`
See also	[005b](#) skips early GP names when starting in a later era. [008a](#) makes minor changes to the wonders available in later start eras. The non-Ancient scenarios of the Dawn of Civilization mod pretend that a certain number of GP have already been born. Git commit: [GitHub](#)
Free buildings per start era: Medieval: Granary, Walls Renaissance: Market, Library Industrial: Aqueduct, Forge Modern: Courthouse, Grocer Future: Factory, Recycling Center	When starting in the Medieval era or later, all cities founded throughout the game receive some free buildings. Medieval: Granary, Lighthouse (if coastal) Renaissance: Aqueduct, Forge Industrial: Market, Harbor (if coastal) Modern: Library, Courthouse, Jail Future: Factory
Rationale	Want to keep it at two buildings added per era. Walls kind of spoil the fun, but Medieval rush tactics seem too potent in BtS. Free Walls in the Industrial era are immediately obsolete but still raise city defense. No more room then for the coastal buildings. I also don't like that these give a somewhat obscure extra incentive (an additional free building) for settling at the coast. Library instead of Forge in Renaissance should help jump-start research, especially through Great Scientists. Aqueduct should stick with Forge as it helps against the health penalty. I suspect Jail was added in the Modern era because espionage was the big novelty when the later-era start option was implemented.
Config	`Civ4BuildingInfos.xml`; to disable Walls (and Dunn) in the Industrial era, set their `MaxStartEra` to `ERA_RENAISSANCE`.
Tbd.	Test this to see if the tech cost modifiers based on start era need to be adjusted.
Decreased the start turn for Medieval and Renaissance start a little.	
Rationale	Should match the tech pace better (or less badly).
Config	`Civ4EraInfos.xml`
Culture level thresholds (except "Fledgling") are reduced when starting in the Medieval era or later. For example, starting in the Modern era halves the culture level thresholds.	The start era does not affect culture level thresholds. There was an unused tag `iCulture` in `Civ4EraInfos.xml` that may have been intended as a multiplier for culture level thresholds.

The victory screen shows how much culture is needed for victory.	Another unused iCulture tag, in `Civ4GameSpeedInfos.xml`, remains unused (but that one is arguably superseded by `Civ4CultureLevelInfos.xml`). The victory screen only says that "Legendary" culture is needed.
Config	`Civ4EraInfos.xml`; I'm using the `iCulture` tag.
Rationale	To make Culture victory viable regardless of the start era. To this end, it would be enough to change the "Legendary" threshold, but might as well change the others too for a smooth progression. Don't change the first threshold ("Fledgling") unnecessarily though; players know that one by heart. Space victory arguably comes too early when starting in the Modern or Future era. This can't really be fixed; players arguably should disable Space victory. Looking at it this way, lowering the threshold for Culture victory may mean that players will also want to disable Culture victory; inconvenient. That said, I don't want to base the culture level adjustment on the assumption that Space victory will be disabled.
Tbd.	I've no idea if halving the Legendary threshold is enough to make a Modern-start Culture victory viable. I haven't tested (AI Auto Play) any of the values so far.
See also	251 adjusts the culture level thresholds to the game difficulty level.

127	Changes to AI Auto Play
Tbd.	It would be nice if AI Auto Play could be interrupted more reliably. No clue how to go about that, "Lock Modified Assets" seems to keep AI Auto Play disabled even after returning to the main menu and starting a new game.
See also	devolution has added benchmarking functionality to AI Auto Play. Some of the Debug mode tweaks (007) are also potentially helpful for all-AI games. CFC post describing the contortions that players have to go through in order to follow an all-AI game without mods.
Credits	The original AI Auto Play mod is by jdog5000, building on earlier work by MRGENIE.

AdvCiv	*AI Auto Play mod*
While Auto Play is enabled, the proxy AI civs (i.e. originally human-controlled) adopt the AI handicap. E.g. in a singleplayer game on Emperor difficulty, the human civ normally incurs 100% inflation, and the AI civs 90% times 80% = 72%. The 90% comes from the AI handicap (Noble), and the 80% from the AI adjustment of the game handicap (Emperor). On Auto Play, the proxy AI civ also incurs 72% inflation.	Auto Play doesn't affect player handicaps; however, proxy AI civs do benefit from AI bonuses defined by the game handicap. In the example, the proxy AI civ incurs 100% times 80% = 80% inflation. Both factors come from Emperor.

Rationale	For simulating all-AI games, proxy AI civs should play by the exact same rules as the normal AI civs. AI Auto Play could also be used for fast-forwarding a normal singleplayer game e.g. when a human victory looks inevitable. In this case, the human AI civ should play by the same rules as a normal human. That's not what the original AI Auto Play does (see example above), and implementing it like that actually looks difficult. For development purposes, it's far more useful to treat proxy AI civs like normal AI civs. This change makes it unnecessary to take the human AI civ out of the game (e.g. by surrounding it with Ice) when running a "Battle Royale"-style AI tournament.
Tbd.	Should scenarios with unequal AI handicap settings (e.g. Earth1000AD) be exempt from this change? E.g. the default handicap of France is Chieftain. If a player plays as France, sets the game difficulty to Emperor and starts AI Auto Play, what should the AI handicap be? Currently, it's the average of the other AI handicaps: Prince. I don't think the difficulty from the scenario is accessible from the DLL if the player configures a different one on the Custom Game screen.
See also	The difference between player handicap and game handicap are also explained under 708.

The proxy AI ignores automation options, in particular "Workers leave improvements/ forests".	Proxy AI civs don't replace improvements and don't chop Forests if the respective options are set.
Minimized popups (from the "minimize-popups" option) are killed when Auto Play starts.	Minimized popups can remain on the screen for some time.
Reminders (Alt+M) are not shown while in Auto Play.	Reminder messages and popups scheduled before starting Auto Play appear during Auto Play.

Rationale	Probably just things jdog5000 hadn't thought of.
See also	This K-Mod 1.45 commit ensures that AI workers don't automate their workers, however, the automation options also affect AI code that gets executed by non-automated workers.

Auto Play ends at the end of the player's turn that precedes the active player in the turn order, i.e. normally at the end of a Barbarian turn. That means, the human civ is treated as an AI civ during the AI turns. In particular, other AI civs can't send diplo messages to the human civ, and thus there can't be diplo popups when human control resumes.	Auto Play ends at the end of a proxy-AI turn. During the subsequent round of AI turns, the human civ is already treated as human. If the human civ is not in slot 0, e.g. in a scenario, it is also already treated as human during the AI turns in lower slots.

Rationale	It's counterintuitive that e.g. running Auto Play for a single turn only skips over the human turn but not the AI turns.
	Diplo messages upon resuming are problematic because they force the human player to make decisions. Makes it impossible to simulate an all-AI game in chunks of e.g. 50 turns without any human influence. More generally, running Auto Play in multiple short chunks produces markedly different results than a single long run; not good.
	Caveat: While there can't be diplo popups anymore right after Auto Play ends, popups are unusually frequent on the next turn. This is because the last-contacted timers of the AI civs keep decreasing during Auto Play. Put differently: The AI has much to say if you haven't spoken in a while.
	For a fair all-AI game on Noble or Prince difficulty, it should now suffice to activate Auto Play on turn 0; for other difficulties, the WorldBuilder is needed for giving the human civ the same initial freebies as the AI civs. Could automate this (special treatment for Auto Play on turn 0), but it seems not quite worth the effort.
Config	A Python change (`AIAutoPlay.py`)
Tbd.	The player can still be forced to make a proposal or vote on one right after Auto Play ends.
	(Not sure if this still applies to the latest version of AdvCiv. Generally, I think there is no difference anymore between a single long AI Auto Play run and multiple short ones.)
See also	044 creates autosaves also at the start of the active player's turn.

The AI attitude cache is updated when AI Auto Play starts and ends and when an autosave that was created during AI Auto Play is loaded.

Rationale	Some of the "first impression" relations modifiers don't apply to humans (i.e. they do and should apply during AI Auto Play).

Disable the Ctrl+Shift+M shortcut.	That combination causes 1 turn of AI Auto Play, billed as "auto moves"/ "moving your units".
Rationale	One turn of AI Auto Play is not the same as letting the AI move the human units – see the notes above about handicaps and the AI Auto Play lasting through the AI turns. It might still be kind of handy to have a shortcut for 1 turn of AI Auto Play, but it shouldn't involve the M key. Ctrl+Shift+1 collides with unit grouping shortcuts, Ctrl+Shift+F1 is taken by the GameFont display ... Anyway, considering that 1 turn is the default duration for Ctrl+Shift+X (a K-Mod change? I think it was 10 turns originally), I don't think a separate shortcut is needed, and it seems safer to always require another click for confirmation before starting AI Auto Play for any duration.
See also	Also want to avoid confusion with the similar key combination for the savemap function. Accidentally running 1 turn of AI Auto Play instead of saving the map would be bad.
AdvCiv	BtS/ K-Mod

Renamed `CvDLLInterfaceIFaceBase::` `addMessage` to "addMessageExternal". Renamed the wrapper for that interface to "addMessage". (Some of the changes described above and below are implemented in that wrapper function.)	Apparently there is a function `CvInterface::` `addMessage` in the EXE, which is responsible for displaying on-screen messages (immediately through the `bForce` parameter, otherwise, the message gets stored at a CvPlayer object). That function is exposed to the DLL through a class CvDLLInterfaceIFaceBase, and gets called in numerous places. The call locations don't check if the recipient of the message is human, which is wasteful and results in a minor memory leak. K-Mod fixes that through a wrapper function `CvDLLInterfaceIFaceBase::addHumanMessage` that discards non-human messages, and changes all call locations from `addMessage` to `addHumanMessage`.

Rationale	Karadoc may not have realized that the it's possible to rename `CvInterface::` `addMessage` despite it being a virtual function. Be that as it may, having a public function called "addMessage" that programmers aren't supposed to use isn't good.
	Discarding the messages to AI players does not avoid the overhead of composing the message string; but that's not a concern because the number of on-screen messages needs to be kept small in any case for usability reasons.

Spectator mode

AdvCiv	*AI Auto Play mod*
When both in AI Auto Play and Debug mode, the human player receives messages about the following major events regardless of whether the civs and cities mentioned in the messages are known to the player's civ: war/peace, Defensive Pact or Permanent Alliance signed, vassal agreement started/ended, city conquered/razed, capital relocated (though not a major event; cf. [106](#)), city culture-flipped, city or colony liberated, wonder completed, Golden Age started/ended, religion founded, state religion changed, resolution of the Apostolic Palace or UN passed, Legendary culture reached, nuke launched/intercepted, meltdown. The messages are shown in the usual colors and can be located on the map once Auto Play ends by clicking on them in the Event Log.	Messages are displayed as if the player was in control, i.e. based on the information available to the player's civ. I think the original Auto Play mod delivered messages as if the human player was in control, including indicator bubbles on the map that weren't cleared until Auto Play ended. No indicator bubbles in K-Mod (good), but colors and location info were also removed.

Rationale	Civics changes (except Free Religion) and random events aren't reported because they occur too frequently.
	It would be nice if messages did not refer to the player's civ as "you" (e.g. "You declared war on Alexander!"), but it's too much work to change this.
	Sadly, one can't use the Event Log to jump to locations that aren't revealed to the active player. The EXE checks `isRevealed` with `bDebug=false` before moving the camera, without calling any DLL code beforehand. (`CvDLLWidgetData::` `doGotoTurnEvent` isn't called either.)

See also	[104v](#) shows messages about AI war plans when in Spectator mode. [007b](#) requires Debug mode for some debug/ cheat key combinations. This should allow players to keep cheats ("chipotle") permanently enabled without using them by accident. [106](#) added the notification about meltdowns for players other than the city owner.
Tbd.	Also show messages about diplo resolutions that don't pass? Known issue: `CvRandomEventInterface.py` calls `CvInterface::addMessage` directly, which means that the code in `CvDLLInterfaceIFaceBase:: addMessage` for suppressing info messages during Auto Play has no effect on messages generated by random events. Would have to expose the DLL's `addMessage` function to Python to fix this, but that's a bit laborious because of all the non-primitive call parameters.

Can use AI Auto Play in multiplayer by setting `ENABLE_AUTOPLAY_MULTIPLAYER` to 1 in `GlobalDefines_devel.xml`. All players become automated and de-automated at once. The K-Mod cheat mode check wasn't implemented correctly. Cheats weren't disabled in multiplayer when a singleplayer game was started beforehand. Now fixed.	Multiplayer is supposed to be supported in the final version (2.0, which is, as far as I can tell, the version included in K-Mod): "*AIAutoPlay is now multiplayer compatible [...]: it will apply to all players, not just the one who activates it.*" ([source](#)) This final addition probably wasn't tested much; due to what looks like a bug, only one player gets automated at a time, and only one can be returned to human control. Moreover, K-Mod 1.30 made this change: "*Autoplay and change player functionality is now only available in cheat mode.*" However, cheat mode ("chipotle") isn't recognized in multiplayer games.
See also	[135c](#) allows Debug mode and WorldBuilder in multiplayer.
Tbd.	Known issues: When AI Auto Play ends in a game with simultaneous turns, the game sometimes goes OOS. Perhaps not difficult to fix. (But not that much of a problem because AI Auto Play with simultaneous turns isn't really useful for OOS debugging anyway – because the RNG sequence isn't reproducible.) Interrupting AI Auto Play through Ctrl+Shift+X stops AI Auto Play only on one machine. Would like to disable AI Auto Play in singleplayer mode when playing with the "Lock Modified Assets" option because AI Auto Play can work as a cheat in difficulties below Noble. However, if it's disabled through `BugEventManager.py`, AI Auto Play will remain disabled after exiting to the main menu and creating a new game without "Lock Modified Assets". So I'd have to disable it somehow in the AI Auto Play component itself.

Start and end of Auto Play are recorded in replays.	
See also	Relevant mostly for [R&F](#)
Civ4lerts are not updated during Auto Play and no alert messages are displayed right after Auto Play ends.	AI civs don't have attached Civ4lerts, but when a human civ comes under AI control during Auto Play, its Civ4lerts keep getting updated. When Auto Play ends, alerts are displayed based on the differences between the current game state and that on the last turn under AI control.

Rationale	The updates during Auto Play are a waste of CPU time. And the game state after Auto Play ends can be completely different from the game state before Auto Play started; alerts shouldn't try to communicate that difference.
See also	[106c](#) prevents alerts from being shown right after loading a savegame.
Tbd.	I've added an is-AutoPlay check to every onEvent function in the two Civ4lert scripts. The cleaner (and faster) approach would be to unsubscribe the Civ4lert objects from being notified of game events when Auto Play starts and to re-register them when Auto Play ends. (It's an Observer design pattern.) My Python skills aren't quite up to this task. (The alerts added by [210](#) are disabled during Auto Play in a clean manner through the `AdvCiv4lerts` class.)
When the active player is defeated during AI Auto Play, the (non-human, non-Barbarian) player with the next highest id becomes the active player. AI Auto Play continues uninterrupted. (If this procedure fails somehow, `checkPlayer` in `AIAutoPlay.py` is used as a fallback.)	When the `checkPlayer` method recognizes that the active player has been defeated, it creates a Settler unit for that player at coordinates (0,0) in order to prevent the game from ending. A popup for selecting a different player is launched and gets displayed at the start of the (defeated) active player's turn. Not sure if that had been working correctly in K-Mod. If it fails, the active may end of with a Lion unit; I guess that happens through BtS code.
Rationale	I think AI Auto Play is mostly used for testing and for watching all-AI games. In those cases, it doesn't really matter which player is the active (human) player, but interrupting AI Auto Play can be annoying when it runs unobserved. It's also difficult to ensure that the original procedure works smoothly.
See also	A bugfix in `CvPlayer::acquireCity` (listed under [001](#)) causes players to be defeated as soon as they lose their last city – rather than delaying defeat until the start of the next game turn. This makes it more difficult to get `checkPlayer` to work correctly and without failed assertions.
When AI Auto Play is used in One-City Challenge or Always War games, those rules continue to apply to the automated human player.	The special rules only apply to players so long as they are under human control.
Rationale	Should help with testing those options through AI Auto Play.

127b	Store location info (`iFlashX`, `iFlashY`) for most announcements

When the player clicks on an announcement in the Event Log, the camera will normally focus on a tile that has something to do with the respective event: Anarchy, Golden Age begun/ ended, project completed, civic/ religion change, circumnavigation: Jump to the capital of the affected civ. DoW, war trade alert: Jump to the capital of the civ that started the war (unless it's the active player) or has become willing to start a war. Peace made, PA, DP, vassal agreement signed/ broken: Jump to the capital of the civ that has initiated the treaty/ cancelation. (One might as well jump to the other capital; this is pretty arbitrary.) Colony liberated: Jump to the location of the new capital. Diplo vote: Jump to the location of the vote source (AP or UN); when a resolution passes, also show an indicator bubble on the map.	Announcements about events that don't happen in a particular tile don't get associated with any tile on the map. Clicking on such an announcement in the Event Log has no effect. Apart from the click-to-jump behavior, announcements with an associated tile will highlight that tile with a flashing dot on the minimap (no change in AdvCiv).
Rationale	No harm in allowing these camera jumps via the Event Log. (The flashing dot doesn't amount to a distraction.) Mostly helpful for following Auto Play games, in order to jump back and forth on the map as wars are declared, ended etc. Indicator bubble for passed resolutions: "This is the city you need to conquer in order to stop such resolutions in the future."

127c	Changes to the ChangePlayer component
Credits	The ChangePlayer mod is originally by jdog5000. I'm not aware of a standalone version. Looks like karadoc merged it into K-Mod; not sure from where exactly – perhaps from the Revolution mod.
AdvCiv	*K-Mod*
Only enable the ChangePlayer shortcuts when the "chipotle" cheat code has been entered.	The shortcuts Ctrl+Shift+P (change the active player's civ or leader) and Ctrl+Shift+L (take control of another player) are always enabled.
Rationale	No cheat code should mean no debug/ cheat/ god mode tools.
Tbd.	Would be better to let a BUG config file (see e.g. `BUG Core.xml`) or `Civ4ControlInfos.xml` handle the shortcuts.
Disable the "update graphics" shortcut (Ctrl+Shift+U).	Not sure what graphics are updated exactly. I don't see any difference. Also, the key combination doesn't work when cheats are enabled because the unit debug menu blocks all combinations that involve Ctrl+U.

Rationale	Since ChangePlayer now requires the cheat code for everything, a different shortcut would have to be used. Ctrl+Shift+G is available. However, it seems that the automatic graphics updates after changing a civ or leader are sufficient. When there is no visible effect (except for a harmless failed assertion from change [210](#)), the shortcut will only cause confusion.
Use a hack that temporarily sets a Barbarian unit flag for every plot in order to update the flag graphics. (Prior to v0.97, I had instead added a `bGraphicsOnly` parameter to `CvPlayer::setCurrentEra` that caused the function to return after `setDirty(Flag_DIRTY_BIT,…)`.)	After a civ change, flag graphics are updated by temporarily changing the active player's era. During the era change, AI Auto Play is enabled in order to suppress the new-era popup.
Rationale	The new hack seems less intrusive. Also allows the flag graphic to be changed separately from the civilization.
See also	Brief discussion about updating flag graphics in the [Realism Invictus thread](#)
No option to change a player's team.	The Ctrl+Shift+P popup allows teams to be reassigned. However, the reassignment is ignored by the code that processes the popup and therefore has no effect.
Rationale	I'm not interested in that feature, so I'm not going to fix it.
Update the AI attitude cache for both civs after switching human control.	No update of the AI attitude cache.
Rationale	Some of the first-impression modifiers don't apply to human players, so the cache needs to be updated.
See also	[127](#) updates the AI attitude cache at the start and end of AI Auto Play.
Merged parts of the `CvGame::nextActivePlayer` function (Alt+Z) into BBAI's `CvGame::changeHumanPlayer` function (Ctrl+Shift+L) because their functionality overlapped. This also seems to have fixed an issue with the diplo screen refusing to put certain trade items on the table when trading with a previously human-controlled AI civ. Apparently, the diplo screen (in the EXE) checks the net IDs of the trade parties, and the BBAI code hadn't updated those. And, in single-player, the DLL no longer relies on the EXE for providing the correct net IDs when loading a savegame from within a running game. It seems that the net IDs stored by the DLL aren't reliable when different player IDs are active in the running game and in the savegame. This could explain why, by my recollection, player cycling through Alt+Z had also been causing the diplo screen issue sometimes (not just Ctrl+Shift+L).	
Tbd.	Perhaps an alternative (cleaner?) fix for the issue with loaded savegames would to be set to -1 for the human proxy during AI Auto Play. (I don't know if the problem had only been occurring with saves created during AI Auto Play.) There appears to be a rare – perhaps unrelated – issue that makes it impossible to add either gold or gold per turn to the trade table. I've only encountered this once in a R&F game, but [this](#) Realism Invictus bug report seems to describe the same problem. While the net ID issue had, technically (at times also visually), allowed trade items to be added but then immediately took them off the table again, the second issue prevents gold from being added in the first place; in the DLL, nothing seems to happen at all once the amount of gold has been entered, in particular no call to `CvPlayer::canTradeItem` or `updateTradeList`.
See also	[CFC post](#) about the player switching and gold trading issues also occurring in RFC mods.

128	AI cheats less obviously with visibility	
AdvCiv		*BtS*
An AI unit can't target units on invisible tiles unless those units are within a randomized subrange of the search range around the AI unit. Units in visible tiles (e.g. visible by a different AI unit) can be targeted within the whole search range; no change.		The Unit AI treats all tiles within a unit's search range as visible.
Rationale	This (well known) AI cheat becomes very obvious when AI ships pursue human Privateers or intercept human cargo. (The search range is a multiple of the number of movement points, and ships have lots of movement points.) Can't just exclude all invisible tiles because this would leave the AI completely unable to pursue units. I also doubt that AI patrols could search for enemy cargo ships effectively. With my change, the AI sometimes finds targets on invisible tiles, and sometimes doesn't. This looks similar enough to human guessing and deduction, which also isn't always successful.	
When checking for possible enemy attacks (plot danger functions), the AI treats enemies in tiles that are not currently visible as if they were visible with a probability of 50%. This is implemented through a hash value salted with the game turn number, meaning that the outcome doesn't change if the plot danger functions are called several times for the same tile on the same turn. When plot danger is checked for human units (e.g. to wake up a threatened Worker), only visible threats are considered (no cheating).		Plot danger is usually only checked within a 2-tile range, so most of the tiles are visible, but this isn't enforced. E.g. if a human Worker builds a road on neutral territory, an approaching Wolf or Panther will cause the game to interrupt the Worker, even if the animal is still 2 tiles away and out of sight.
Rationale	Same as above.	
See also	001k fixes a bug in the plot danger functions. 001i prevents the plot danger functions from taking into account roads in the fog of war.	
When checking if human units are in danger, the pathfinder is used for greater accuracy. This mostly (only?) applies to Workers getting interrupted. Danger from more than 3 tiles away is ignored. During the first two eras, the AI also uses the pathfinder but only when the potential threat is a human unit.		Same treatment for human and AI: To decide whether an enemy unit (on the same landmass) is a threat, the air distance is compared with the movement points. If the unit is standing on a road or railroad, its movement points are treated as 1 greater. Terrain and whether the route connects to the target isn't checked, so false positives can easily occur. Will not notice possible attacks by units with terrain movement bonuses (e.g. Woodsman II promotion).
Credits	Leoreth brought up the issue of false positives here on CFC.	

Rationale	See the link above. I'd like to use the pathfinder always, but, at longer distances, this could give away information about fogged or unrevealed tiles along the way, and the plot danger functions are called very frequently by AI code. In profiler runs, using the pathfinder for the AI (when the air distance is 3 or less) added something like 5% to the total runtime, which is clearly not worth it. The pinpoint change to address early attacks by human Woodsman units on AI Workers isn't going to affect performance.
Tbd.	Perhaps increase the 3-tile limit to 4.

The tactical AI for air missions uses perfect information only sometimes (randomly) when checking for enemy air interceptors.	Always ignores the fog of war when checking for enemy interceptors.
Rationale	At least for the paradrop mission, playing fully by the rules would seem too easy for human players to take advantage of. The AI could also look bad with Airships against a sparse human Fighter force. Cheating always, on the other hand, can be noticed. (Though I think few players if any have ever noticed. So I wouldn't really mind simply reverting this change if it turns out to hurt the AI significantly.)
See also	Based on code written for 004c (enemy interception chance in help text).

129	Changes to resource, feature and river placement during map generation
See also	021b prevents Fur from being placed on hills and allows Banana, Sugar and Ivory on Plains Jungle. tsl makes some changes to the Fractal landmass generator. 108 makes changes to starting site normalization 191 changes how random civs and leaders are chosen

AdvCiv	*BtS*
Place fewer copies of resources that have a lot of eligible tiles, and thus would normally be placed abundantly. E.g. a bit less seafood on maps with lots of coastline, and sparser resources overall on Huge maps (but still far more in total than on Large maps).	The number of copies of a given resource is proportional to the number of tiles where the resource could be placed.
Rationale	Make unusual maps a little less unusual and more playable.
Config	`SUBLINEAR_BONUS_QUANTITIES` in `GlobalDefines_advc`
When placing (strategic) resources based on the map's player count, the number of resources to be placed is increased a little bit on maps with fewer than 8 players and decreased a bit on maps with more than 8 players. E.g. on a map with 16 players, the number of resources placed is the same that BtS places for 14 players.	The number of resources placed for the `iPlayer` XML tag increases linearly with the number of players alive.
Rationale	On overcrowded maps, strategic resources can become conspicuously commonplace. Should be OK to place fewer copies when there are more players from whom to capture or snatch away resources.
Tbd.	There isn't a strong reason for applying this change to large maps that aren't overcrowded. So perhaps it shouldn't apply to such maps.

When placing a cluster of resources, the probability of placing an additional resource decreases exponentially with each resource already placed. The potential target plots are processed in a randomized order.	The targets are processed in clockwise order, and each one gets the same probability (25 or 50%, depending on the resource) until the upper limit is reached.

Rationale	Make large clusters less likely, especially on maps that have large areas of uniform terrain because such areas are especially likely to receive large resource clusters (e.g. Ivory in a large area of Plains).

The map generator avoids placing more than two resources of a kind adjacent to each other, in particular when placing a cluster of resources on a 3x3 square.	It seems that a cluster can theoretically fill an entire 3x3 square. The number of copies to be placed on the map in total puts a cap on this, but not on large maps.
Added a separate resource class for Gold, Silver and Gems. Clusters of these resources are now kept far enough apart to prevent a single city from working all resources in both clusters. (Actually, Silver can't be placed in clusters anymore at all; see a few boxes below.)	Nothing to ensure that multiple clusters are placed apart.
Clusters of Gold, Gems and Ivory get spread out over 21 tiles (a "city cross"); individual Gold and Gems resources can't occur adjacent to each other within that range. Decreased the placement probability so that the number of resources in a cluster is, on average, smaller than in BtS (despite the increased diameter). Still not rare to get more than 1 Gold or Gems within a city's radius, but 3 or more are quite rare now, even on large maps.	All clusters get spread across a 3x3 square. The map generator can produce large clusters of precious metals/stones that make for very powerful (starting) city sites. 5 such resources aren't that uncommon on large maps.

Rationale	Starting plots with more than one Gems or Gold tend to be overpowered. Could also be fixed when assigning starting plots or during normalization; however, the cluster placement algorithm also scaled badly with map size, so it needed work anyway.
	Ivory: Large clusters aren't really useful and look a bit strange; let's give the jumbos some more room. (So long as War Elephants don't become too widely available this way … If that's a problem, then it could help to restrict Ivory placement during the normalization of starting sites.)

See also	108 never removes bad terrain or features from precious resources when normalizing starting locations. Also weakens starting locations overall, and increases the value counted for commerce a bit when evaluating starting locations.

Fur, Gold, Silver and Gems can appear on multiple landmasses.	Fur, Gold, Silver, Gems, Banana, Corn, Cow, Deer, Pig, Rice, Sheep, Wheat, Dye, Incense, Ivory, Silk, Spices, Sugar and Wine (19 out of 35 resources) get placed only on one continent.

Rationale	The single-continent restriction doesn't match the distribution of those resources on Earth at all. As for encouraging trade, Fur isn't really a concern because it often appears in groups. The precious minerals at least sometimes appear in groups and their terrain restriction also tend to cluster them a bit.

Tbd.	For Spice, Wine, Incense and Dye, the single-continent restriction is also dubious, and they also appear mostly in clusters. (The restriction is defensible though by arguing that e.g. Spice represents only black pepper and other spices of southern Asia; that Wine refers only to Old World vines; Incense only to frankincense; and Dye only to indigo.) Perhaps just remove the restriction for Wine, Incense and Dye; that would still leave more than 1/3 of the resources overall unique to one continent.
Config	`bArea` in `Civ4BonusInfos.xml`

Relaxed the latitude restriction for Crab.	Crab can only occur at latitude above 40°.
Decreased the overall frequency of Fish a little.	Fish tends to be the most common resouce.

Rationale	Too few Crabs; was quite a bit more rare than Clam. A latitude restriction for Crab isn't realistic at all; they're caught all over the world. That said, keeping the range of Crab and Clam at least somewhat apart is arguably good for gameplay.
	Decreased Fish frequency to compensate for the Crab change. As an Ocean resource (potentially), it also has too much impact on city placement.
Tbd.	Would like to restrict the latitude range of Fish. That would be realistic insofar that, historically, mainly fish from the North Atlantic was used as a (long.distance) trade commodity. (On the other hand, fish was a major food source for the Norte Chico civilization near the equator ...)

Livestock resources can only be placed at a latitude of at least 9°; Sheep: 25°.	No lower bound for Horse and Cow. Pig had required at least 10°, Sheep 30°.

Rationale	To account for the tsetse fly. That only affects Africa, but there also aren't a lot of large farm animals in equatorial Brazil, Malaysia and Indonesia (though there probably are some regions there with plenty of water buffalo or pigs). 9 or 10° makes a difference on Standard-size maps (at least on Fractal). 9° only excludes livestock from 4 map rows, 10° from 6 map rows.
	Sheep are commonly kept in southern Iran and northern India, so 30° is a bit too strict.
Tbd.	Equatorial "jungle" (rainforest) is still far more productive than it should be.

Horses can be placed at a latitude of 9° (see above) to 68°.	0-90°

Rationale	Wikipedia lists 66.9°N (Finland) as the world's "nothernmost Equine veterinarian". I suppose Yakutian horses probably also aren't kept farther north than that.
	The restriction won't matter much as there isn't much Tundra beyond 68° latitude.

Reduced the unique range of Horse and Oil a little bit.	There have to be at least 6 tiles in between two Horse resources and 7 in between two Oil resources.

Rationale	Understandable for gameplay reasons, but quite unrealistic. Now a little less so. The steppe belts of the Earth are vast grazing grounds for horses.
Tbd.	I have some plans for making (native) access to Oil less crucial. Eventually, Oil should actually form clusters.

Map scripts try to place at least 0.78 Silver resources per player, and only 0.78 Gold resources. Gold can appear on Snow, Desert and unforested Plains, river possible, whereas Silver can appear on forested Grassland and on Tundra regardless of Forest, but not along rivers. Silver can't be placed in a cluster. Adjusted AI tech value so that Bronze Working is prioritized when starting next to a Silver Forest.	1 Gold per player and 0.67 Silver. Gold only on Desert and unforested Plains, and Silver on Snow and unforested Tundra. Both can appear next to rivers.

Rationale	Gold was arguably more rare throughout most of history. More importantly, Gold is very powerful, and fewer occurrences should improve the game balance a bit. That said, Grassland Silver Hill is nearly as powerful as Desert or Snow Gold Hill. Therefore I haven't made Silver more abundant than Gold; don't want to make it too much more important than in BtS. Allow Silver to appear on forested Grassland to make it more easily accessible. Also, restricting Silver to extreme latitudes is, as far as I can tell, completely unrealistic; should really appear on just the same tiles as Gold. The intention was probably to make Gold and Silver more distinct. I'm trying to keep them distinct but make the distinction more historically informed. The most famous gold rushes happened in the Australian desert, Alaska and maybe "on the banks of Sacramento," as a shanty says. Silver can't be or isn't commonly panned from rivers. The densely forested Central European mountain ranges have seen silver rushes in the Early Modern era (Berggeschrey). Grassland Forest also fits with Colorado and British Columbia. Indeed, silver mining might be more reliant on charcoal for refinement than gold mining. Ancient Egyptian gold deposits and silver in Attica and Baetica also fit. Silver clusters disabled because, on Grassland Hills, they can be pretty powerful and can easily occur near starting sites. Gems can only appear on Jungle Grassland; thus distinct from Silver.
Credits	Inspired by Civ 4 Reimagined 1.2, which also makes Silver more common than Gold.
See also	The BASE mod also allows (my translation) "Gold in cold areas and Silver in warm areas" – though I didn't get the idea from there. rom3 enforces terrain restrictions also when Mines make random discoveries. As Mines remove Forest and Jungle, feature requirements are not enforced.

When the same placement priority is set for two types of resources in XML, decide randomly which type to place on the map first. Similarly, during starting site normalization, go through the resource types in a random order when looking for a suitable food or extra resource.	BtS relies on the order of the XML file (mostly alphabetic) in these cases. That results in a bias for certain resources near starting locations and causes Sugar (one of the last resources in the file) never to be placed on extremely crowded maps.

See also	Prompted by this CFC thread about certain resources never appearing on very crowded maps.

Rationale	Having a great variety of starting locations (or rather – among balanced and historically plausible starting locations) is pretty important, so arbitrary biases should really be avoided. With my changes, some resource type may still be omitted on extremely crowded maps, but it won't always be the same type.
See also	<u>advc.enum</u>: The `FOR_EACH_ENUM_RAND` macro makes it pretty painless to randomize the order in which alternatives are considered. During normalization, <u>108</u> also avoids placing resource types that already exist around the city.

When generating river sources, ties in the order in which BtS considers tiles are broken randomly.	For rivers sources, certain tiles are prioritized, especially hills and peaks, but e.g. among all hill and peak tiles, those with the lowest map index are tried first.
Rationale	Should increase the variety among generated maps a little.
When deciding where to place a river segment, all four tiles (or rather: their terrain types) at whose corner the segment will arrive are given equal weight.	Rivers tend to flow away from peaks and hills and toward flat land and water. The terrain of the tile at whose southeastern corner a segment will arrive is given much higher weight than the surrounding tiles. This gives one riverbank more weight than the other and sometimes gives the terrain orthogonally adjacent to the river segment more weight and, at other times, the terrain diagonally ahead of the segment.
Rationale	Probably just laziness on the part of the Firaxis developer; obviously both sides of a river should be treated equally, and the terrain next to and ahead of the river should always matter, or at least it shouldn't make a difference whether the segment goes west-to-east or south-to-north etc.

Increased the attraction of rivers to water tiles a bit.	
Rationale	Should result in somewhat shorter rivers overall. BtS generates very long rivers too frequently for my taste.

Added a limit for the number of resources of one type to be placed on a landmass: Can only place up to 2 + ((number of tiles on the landmass) / 25). Excess resources are placed elsewhere; that can include small islands.	When placing a resource with a single-landmass restriction, as many resources are placed as there are valid locations. Resources that can't be placed on the target landmass aren't placed at all. The target landmass is chosen randomly with a bias for large landmasses. K-Mod has added a clause that rules out small islands entirely.
Rationale	Mainly so that single-landmass resources can also appear on small islands. E.g. the Spice Islands hadn't been possible with the BtS/K-Mod rule.
When selecting a landmass for a resource that is only supposed to appear on one landmass, take into account how many tiles on each landmass are eligible for receiving the resource.	Only based on total landmass size and the number of single-landmass resource types have already been assigned to each landmass.
Rationale	Especially Banana was sometimes placed quite densely on landmasses with very little Jungle. Can still happen though.

Decrease the variance in the number of Banana and Fish resources placed.	The number of resources placed is based in part on player count (fully predictable), in part on the number of tiles eligible for receiving the resource (somewhat unpredictable). Further variance is explicitly added through the `RandApp` (random appearance) XML elements.
Rationale	See above about Banana; having an especially high number can look bad. I think having very few Bananas sometimes isn't very interesting either. Similarly, neither abundant nor scarce Fish plays particularly well.
Tweak the calculation of per-tile latitude values in the DLL: The maximal latitude value (normally 90) is assumed to refer to the top edge of the map and the minimal latitude value (normally -90) to the bottom edge. The latitude values of tiles are interpolated based on this. Consequently, e.g. the latitude value of the topmost row will normally be slightly less than 90.	The topmost row gets the maximal latitude value, etc.
Rationale	The game shouldn't suggest that a whole row of tiles exists at 90° or -90° latitude. This change gives all latitudes that aren't close to 90 a tiny bit more space; should have only minimal impact on the terrain and resources that get placed.
Tbd.	A whole row for e.g. 87° latitude is still a gross distortion, but it's really up to the map scripts to set more sensible latitude limits. So – I could do that; should then also make scripts like Fractal and Pangaea place more land near the edges of the map.
129b	No flood plains on river bends
Flood Plains appear only on Desert tiles that are orthogonally adjacent to a river. Desert tiles that are only diagonally adjacent to a river still get 1 commerce from the river but no Flood Plains feature. ~~Oasis can appear diagonally adjacent to a river (but is no more common there than on non-river tiles).~~ This change affects all standard map scripts and also [PerfectMongoose](#).	Also on river bends and tiles diagonally adjacent to a river mouth. Oases can't be adjacent to rivers.

Rationale	Cities with too many Flood Plains don't play well, and, in reality, not every flat desert soil becomes fertile when irrigated. This could be addressed by reducing the appearance probability of the Flood Plains feature from 100% to, say, 67%. I like the river-side rule better because, graphically, Flood Plains on river bends can be difficult to see and bending rivers are the ones that produce the highest number of Flood Plains within a city radius in BtS. I've thought about an exception to allow Flood Plains diagonally adjacent to river mouths as the flood plain is usually widest at the mouth: 　　　□ 　　□‖□ But this really isn't a good way to depict a river delta; should use distributaries instead. Also, the absence of a Flood Plains diagonal to the river mouth leaves room for a city and thus encourages civs to settle at the river mouth. Added an XML tag to allow Oases on river corners so that Desert river corners aren't always useless, and because oases near desert rivers seem pretty common in reality (e.g. Faiyum, Merv/ Murghab, Tarim). Left the tag unused after realizing that a 3-commerce Oasis can be quite powerful at a starting location, especially with the Financial trait.
Config	`Civ4FeatureInfos.xml`, `PerfectMongoose.py`
Tbd.	Shorten the Flood Plains texture so that it doesn't get so close to the tile corners. Currently, Deserts on river corners look (almost?) the same as in BtS, so it's easy to forget that corner tiles no longer receive Flood Plains. The 1 commerce on river corners is pretty pointless. Could give the Desert terrain +1 food next to river through XML (`RiverYieldChanges`). That would only apply when no feature is present, and a hill would eliminate the 1 food, so it wouldn't stack with anything. However, there aren't always adjacent Flood Plains, and, without those, the graphics look completely dry. Such dry stretches do exist in reality as well, e.g. along the Nile cataphracts separating ancient Egypt and Nubia. At least for scenarios, barren desert river needs to be possible. Perhaps I should revert the change to the map generator, i.e. again place Flood Plains on river corners, but grant only +1 food there. This would require a new XML tag though, say, "RiverSideYieldChanges". Would be nice if the Fractal map generator would sometimes create distributaries at a river mouth. I think, currently, this can happen but is very rare.
Config	`Civ4TerrainInfos.xml`

AdvCiv/ BtS	Unofficial Patch
Founding a city on a Flood Plains causes the Flood Plains to be removed forever (just as any other terrain feature). If the city is razed, an empty Desert tile remains.	Founding a city still removes all features, but, when a city is razed, a Flood Plains is placed in the city tile if a Flood Plains can be placed there. Comment by jdog in `CvCity.cpp`: "*replace floodplains after city is removed*"

Config	Can enable the code from the Unofficial Patch through `GlobalDefines_advc.xml`.
Rationale	Don't want to encourage players to raze cities that were founded on Flood Plains just so that the Flood Plains can be worked. Tedious micromanagement. In scenarios, the Unofficial Patch may "re"place Flood Plains where none have ever existed. And players may get the impression that Flood Plains aren't removed in the first place and get confused about why founding on a Flood Plains doesn't yield extra food in the city tile. The disappearance of a Flood Plains also isn't necessarily more implausible than the disappearance of other terrain features: The citizens may well have engineered the river or degraded the soil so that it's no longer arable.

129c	Climate/ terrain adjustments (for all standard map scripts)	
See also	021a allows Jungle to appear on Plains	
Config	All changes below can be disabled by setting `bEarthlike` to `False` at the start of `CvMapGeneratorUtil.py`. They're automatically disabled when playing with a map script that overrides any essential method of the standard terrain generator. From among the official and bundled scripts, the following are not affected by the AdvCiv changes: Rainforest, Highlands, Earth2, Donut, Boreal, Arboria, Fantasy, Great_Plains, Hub, Oasis, Ring, Wheel, Ice_Age, RandomScriptMap, Team_Battleground. (I've considered tying the changes to a new climate type, but it wouldn't have been flexible enough for what I wanted to do. Also, my changes are more subtle than those of the existing climate types.)	
AdvCiv		*BtS*
No bias for Grassland over Plains near the equator (unless a map script sets a custom `fGrassLatitude` interval greater than the default value). Slightly increased the frequency of Plains overall and decreased the grain value, meaning that Plains tend to clump together a bit more than in BtS. Deserts have an (explicit) tendency to appear next to plains.		The Python TerrainGenerator ("*the only primary method for generating terrain types*" according to a comment in `CvMapGeneratorUtil.py`) reserves the +/-10 latitude region around the equator for Grassland. Map scripts can set a different latitude interval (not sure if any do). Since Deserts don't appear near the equator and tiles near the equator are 100% Grassland, Deserts appear more commonly next to Plains than one would assume based purely on the overall terrain frequencies.
Rationale	Plains seem like a slightly more realistic representation of soils with jungle or rainforest cover than Grassland. Such soils are usually not well suited for food production. As for game balance, I don't think it's a fundamental change: Grassland is generally a bit better than Plains, but 100% Grassland can easily result in too little production. The change also makes maps more varied. Additional Plains near the equator imply fewer Plains elsewhere. Since there was already a lack of anything resembling the Earth's steppe belts, I've increased the Plains frequency.	
See also	Experience with the PM script (021b) suggests that steppe and desert belts of realistic proportions disadvantage nearby starting locations too much – unless more sophisticated code is written for the selection of starting locations.	
Tbd.	I have smarter code for starting locations now (027). So perhaps the standard terrain generator could take a few more leaves out of PM's book now?	
Increased the frequency of hills a tiny bit.		
Rationale	Don't want to change the balance too much, but more non-arable land would feel more realistic.	

129d	No unreachable resources	
AdvCiv		*BtS*

Water tiles that could only be worked from a city on an impassable tile (Peak) are counted as unworkable, meaning that they have 0 yield and no resources can be placed there.	All water tiles that aren't within a city radius around any land tile are considered as unworkable, have 0 yield and can't receive resources (well ... map scripts can ignore this). Doesn't check whether a city could actually be founded on the land tile.
Rationale	Just makes players regretful to see unworkable resources. Could've removed them after map generation, but it turns out that it's actually easier to prevent them from being placed.
	On a related note: Unworkable tiles having 0 yields means that players can tell from the yields of an Ocean tile whether there is land nearby. If an Ocean tile is workable, there has to be an adjacent Coast tile. I'm not sure if such a Coast is always visible via graphical bleed anyway, or if the yield display actually leaks information. Wouldn't be easy to fix.
Tbd.	Not covered: Water tiles workable by a city - but fully enclosed by Ice and thus not reachable for Work Boats. Such tiles shouldn't receive resources, but that's not so easy to prevent.
See also	Brief exchange about unworkable seafood on CFC (toward the end of the post): link

130	Changes to AI diplo modifiers (Dynamic Diplomacy)
See also	112: vassal agreements (especially voluntary ones, whereas 130v and 130d focus on capitulated vassals) 141: No diplo effect from gifted GP 553: "Shared discoveries" relations modifier
Config	Some XML values defined in `Civ4LeaderHeadInfos.xml` get modified in the DLL. See comments in Alexander's section of the XML file.
AdvCiv	*BtS*
130a	Sustained peace
"Years of peace" only start to count when an AI civ is met. The turns needed per relations bonus decrease from 60 initially to 30 in the middle of the game (normally AD 1700) and then stay at 30.	Years of peace are counted from the beginning of the game. The relations bonus is one for every 60 turns (and at most 2).
Rationale	Being ignorant of each other's existence shouldn't "strengthen relations," and civs that have just met shouldn't have strengthened relations. That said, don't want to make it too difficult for civs that start isolated to find partners on other continents. The decreasing threshold should also help keeping "years of peace" relevant after war in the late game – when the game may end in 100 turns, it's almost irrelevant whether relations are going to improve 60 turns from now.
See also	130k adjusts this counter to the game speed setting.
130b	Personality-based modifiers

Peace weight reduced to 45% and warmonger respect to 75%, resulting in diplo from peace weight between -2.7 and 1.8, warmonger respect between 0 and 1.5 and their rounded sum between -3 and 3. To this sum, a modifier from handicap is added (see 148) and a base modifier (e.g. Gandhi +2, Tokugawa -1; no change).	Diplo modifier from peace weight between -6 and +4, warmonger respect between 0 and 2. (The peace weight of an AI leader is apparently a measure of that leader's love of peace. It's partly randomized at game start. Leaders with similar peace weight like each other, and leaders with dissimilar peace weight dislike each other. Warmonger respect says whether a leader is considered to be a warmonger, and warmongers like each other.)
Rationale	I think these modifiers are intended to prevent warlike leaders from fighting inconclusive wars among each other while the peaceable leaders get ahead economically. Fair enough, but it's overdosed, sometimes leading to practically inseparable bonds between peaceful civs. The excessive modifiers make diplomacy overall too predictable from the beginning.
Tbd.	When there is a large majority of either warlike or peaceable AI leaders in a game, the small minority doesn't stand much of a chance. Can easily happen when AI leaders are chosen at random. It's less of a problem with the change above, but I'm still considering to adjust the hidden modifiers based on which leaders are in the game. Could adjust the modifiers versus all AI leaders that are too popular or too unpopular overall. Don't want every game to have perfectly balanced diplo though ... Another problem: Don't want the modifier to give away info about the leaders that the player hasn't yet met. If only met civs are taken into account, the first impression modifier may change upon meeting a civ. But "a first impression is a lasting one" suggests that the modifier doesn't ever change. Could I bias the random leader selection instead? Perhaps gradually lower the first impression modifiers throughout the second half of the game: brief CFC discussion
See also	Explanation of Civ 6's "first impression of you" modifier (only conjectures as the source code isn't released): CFC post

130c	Rank-based modifiers	
AdvCiv		BtS
The AI dislikes civs that are ranked higher on the leader board but not those ranked far higher.		Civs on the bottom dislike those on top; the greater the rank difference, the greater the hate. The second doesn't dislike the first, but the last does.
No rank-based modifiers in the first game era.		Modifiers change a lot in the Ancient era because ranks change a lot.
The AI assumes that civs not yet encountered are ranked on the very bottom (or top) of the leaderboard; i.e. these civs don't affect relations.		The AI magically knows the ranks of unknown civs. Humans can sometimes use inter-AI diplo values to deduce the ranks of unknown civs.
Example: Assume that the human player is ranked 7th at the start of the Classical era, and gradually climbs to rank 1 over the course of the game. AI Peter starts on rank 4 and stays there until overtaken by the player. Let's say he meets		+1 until the player reaches rank 3, then 0, and -1 when the player takes rank 1. I suspect that the initial +1 is due to a bug and should be +2 until the player reaches rank 6.

	all his rivals during the Ancient era. Peter's modifier toward the player then starts at +1, changes to 0 when the player reaches rank 5, to -1 when Peter is overtaken, -2 when the player reaches rank 3, -1 again at rank 2, and 0 as the player takes rank 1; see the table below.	
Rationale	Presumably, a (slight) tendency of the AI to gang up on the leader of the field was intended. Not unreasonable, but it doesn't work: Civs that are far behind can't do much to impede the leading civs. They just hurt themselves by sulking.	
	It's not necessarily wise for civs in the lower half to stick together either. Cooperation is just one way to improve, war another.	
Tbd.	Should be based on score ratio instead of rank difference.	
See also	Rank hate can oscillate in the early game (and my changes may have made this worse). 148 alleviates this problem a bit by lowering the relations threshold for worst enmity (i.e. making the AI more hesitant to single out a worst enemy).	
Civs in the lower half of the leaderboard no longer have +1 relations with each other. However, certain leaders still grant a relations bonus to civs ranked beneath them (no change), and rank-based penalties are reduced among civs in the lower half. They're only slightly reduced though so long as relatively few civs are known.		
Rationale	It's not necessarily wise for civs in the lower half to stick together either. Cooperation is just one way to improve, war another. That being said, there mustn't be high penalties for civs that are themselves struggling to keep up.	
Tbd.	Perhaps peaceful leaders should grant the bonus after all (as proposed by CFC user Inthegrave here). Though not in the Ancient era.	
Willem dislikes civs that are ahead of him and likes civs that are behind him, just like all the other leaders.		Willem is the only leader that dislikes civs that are behind him, and likes those that are ahead.
Rationale	Probably a mix-up by the BtS developers.	
No diplo bonus from rank when the low-ranked civ is already at stage 3 or 4 of a victory strategy. Conversely, AI civs at stage 3 or 4 of a victory strategy only assign a diplo penalty to civs at victory stage 4.		AI victory strategies were introduced by BBAI; the diplo modifier is only based on rank.
Rationale	In the endgame, score and rank become unreliable measures of how well a civ is doing.	

Rank player vs. Peter	7-4	6-4	5-4	4-5	3-5	2-5	1-5
Modifier AdvCiv	+1	+1	0	-1	-2	-1	0
Modifier BtS intended(?)	+2	+1	+1	+1	0	-1	-1
Modifier BtS bugged(?)	+1	+1	+1	+1	0	0	-1

Example 130c

130d	A master can't be its own vassal's worst enemy, and capitulated vassals can't be anyone's worst enemy. Can't ask civs in a vassal-master relationship to stop trading with each other. If a vassal has either a forced (by the master) or unforced attitude of Cautious or better toward a rival, then that rival can't be the vassal's worst enemy.
Rationale	Most of the consequences of worst-enmity don't apply to the master anyway, in particular, the vassal will always trade with the master. That means, a vassal with its master as worst enemy effectively has no worst enemy at all. Perhaps OK, but certainly don't want a vassal to promote trade embargoes against its master.
	UWAI ignores relations with capitulated vassals; to be consistent with this, capitulated vassals should not be worst enemies. Capitulated civs are already marginalized; their enemies should be content with this.
See also	The Dawn of Civilization mod also rules out worst enmity when the rank difference between two teams is big. Git commit
	130p prefers war enemies as worst enemies; 148 changes the relations threshold for becoming worst enemies.
130e	Worst enemy updated upon relations change

AdvCiv	*BtS*
Whenever the relations value of an AI civ changes, that civ's worst enemy is immediately updated.	Worst enemy is only updated at the end of an AI team's turn, i.e. never during a human turn. Attitude is updated immediately though, so, e.g. after making peace with a human civ, if the AI attitude changes from Annoyed to Pleased, the AI will still consider the human its worst enemy for the rest of the human turn.

Rationale	Attitude and worst enmity should be consistent during human turns.

130f	Trade embargo changes
See also	130m: AI requests an embargo when at war and too Annoyed to ask for military aid. 001: Bugfix in the computation of the price charged for an embargo. 001e fixes a bug that causes the AI to make a stop-trading demand against a civ that has just stopped being its worst enemy. 104o uses the AI embargo trade value as a lower bound for the AI war trade value.
Tbd.	Would be nice to let the Foreign Advisor show the time-to-cancel for deals between other civs. Would have to happen in `CvGameTextMgr::getDealString`.
	A dozen AI leaders have stricter attitude thresholds for embargoes than for sponsored war (Alexander, Brennus, Hammurabi, Mansa Musa, Mehmed, Peter, Roosevelt, Saladin, Suleiman, Suryavarman, Wang Kon, Zara Yaqob; maybe more when `THEM_REFUSE_ATTITUDE` is counted as well). E.g. when Mansa Musa is Pleased or Cautious, he can be persuaded to begin a war but says "We don't like you enough" when it comes to embargoes. He likes to trade, and I guess war is seen as a trade in this context (hired for war) and embargo as sth. that severs trade; but of course an embargo is also a kind of trade and a hired war also severs trade ... These leaders should perhaps be willing to stop trading when willing to go to war, but only for a high price.
	An alert (see 210) about AI willingness to stop trading would be helpful.
AdvCiv	*BtS*

Apart from vassal and peace treaties, a trade embargo cancels all deals, including those recently signed. The AI refuses to stop trading if it is currently receiving reparations from or paying reparations to the target: "Not right now ... Maybe we'll change our mind in a few years."	Embargo doesn't affect deals with a positive number of turns left to cancel.

Rationale	It's [confusing](#) when some deals aren't canceled, especially between AI civs because players can't see which inter-AI deals are recent. This contributes to embargoes being largely irrelevant, and allows players to undercut AI embargo requests by renewing resource trades every 10 turns.
Tbd.	Or rather treat reparations as in BtS?

Can't propose a trade embargo if currently trading with the target. Once enacted, the civ that proposed the embargo is affected by it as well, i.e. the target won't talk to the instigator, though there is no diplo penalty for having "stopped trading with us", only the penalty for negotiating a trade embargo.	Except for the diplo penalty (negotiated a trade embargo), the embargo does not affect the civ that proposes it.
Exception: A master can always tell its capitulated vassals to stop trading; this does not disrupt the trades of the master. Moreover, the capitulated vassal agrees to the embargo without asking for compensation. The master still receives a diplo penalty from the embargo target though ("negotiated a trade embargo").	The vassal asks for compensation.
When a capitulated vassal breaks free or when a civ capitulates, all stopped-trading memory of and about that civ is deleted.	If e.g. a master asks its vassal to stop trading with a rival, the rival continues to not talk to the vassal even if the vassal breaks away.

Rationale	Trading with a civ after negotiating an embargo seems implausibly hypocritical. The precondition for proposing an embargo (having no current deals with the target) is needed because, otherwise, embargo proposals could be employed to terminate uncancelable trades at will. This would be problematic in the case of per-turn war reparations. Regarding the exception for capitulated vassals: Colonies being allowed to trade only with the mainland was a common practice in mercantilism. I'm keeping the diplo penalty because I don't want players to routinely enact embargoes after accepting capitulation; that would be a bit tedious.
Tbd.	Should embargoes affect all players on the team that agrees to stop trading? Currently, due to `CvPlayer::isTradingWithTeam`, embargoes only affect all team members on the target side.

Stop-trading resolutions (UN, AP) cancel recently signed deals, and such deals don't prevent the resolution from being proposed. Exception: The civ that proposes the resolution mustn't have deals with the target that are too recent to be canceled. The other AI civs try to honor their commitments by voting against the resolution if they have deals with the target that couldn't normally be canceled.	Recently signed deals don't prevent resolutions, but resolutions also don't cancel such deals.	
Rationale	Mostly to prevent human players from gifting 1 gold per turn to an AI in order to block stop-trading resolutions for 10 turns and to prevent human players from proposing a resolution that cancels tribute or reparations.	
See also	Consistent with kekm.25: war resolutions overrule peace treaties, but the proposing civ mustn't have a peace treaty.	
When a player proposes a trade embargo as part of a peace treaty, the AI doesn't refuse on account of attitude. Attitude (toward the war enemy and toward the target of the embargo) still factors into the trade value that the AI assigns to the embargo.	Normally refuses: "We don't like you enough" or "we couldn't betray our close friends".	
Rationale	Makes sense to ask the losing side in a war to stop trading e.g. with remaining war enemies of the winning side, and the losing side should be open to this. Similarly, the BtS AI doesn't check attitude when asked to change its religion as part of a peace deal.	
See also	Similar issue with war trades as part of a peace treaty; see 100. Similar rationale for 132 (change civics as part of a peace treaty).	
Tbd.	The AI should demand this when Furious and at war with the target or when the target is the worst enemy despite the ongoing war. 039 might have to be adjusted.	
When asked to stop trading, the AI assigns greater trade value to OB while at war. Exception: trade value not increased when the civ that asks for the embargo is at war with everyone that the contacted civ is at war with.	OB doubles the trade value of the embargo, regardless of war.	
Rationale	OB are potentially more useful at war. Canceled OB can prevent the AI from reaching its target cities. Unlikely to be a problem when the civ who's asking is a war ally.	
Tbd.	A proper evaluation of an OB agreement while at war would be too much work to implement. But the AI should generally be very reluctant (possibly refuse) to sever OB while at war and also while preparing war. The latter part is problematic because it could expose AI war preparations.	
Trade value charged by the AI for an embargo is reduced by 25% if Pleased toward the (human) civ that asks for the embargo, by 50% if Friendly.	No impact of attitude toward the civ that pays; only the attitude toward the embargo target matters.	
Rationale	Embargoes seemed slightly overcosted overall (now that the preconditions are stricter), and both attitude values should matter. At Annoyed attitude, the AI refuses embargo trades anyway; therefore no point in a cost increase when attitude is low.	

Refuse-to-talk duration after embargo is 18 turns on average. The trade value charged by the AI is proportional to this expected duration (apart from other factors).	30 turns
"You agreed to stop trading with our worst enemy" is remembered for 50 turns on average.	100 turns

Rationale	30 is too long considering that diplomacy is now more dynamic overall. And can let the AI offer an embargo at a more reasonable cost now that the duration is shorter.
See also	130r makes the embargo target forget that the player stopped trading after 60 turns on average. (The BtS AI never forgets this.)
Tbd.	Should perhaps use the same AI memory decay rate for "You agreed to stop trading" as for "You stopped trading." Not sure if 50 or 60 would be the better expected duration though.
Config	`Civ4LeaderHead.xml`; will have to change the values for each of the 52 leaders though.

The relations penalty for negotiating a trade embargo is normally remembered for an average 60 turns (no change), but an embargo proposed against a war enemy is forgotten twice as fast by that enemy (i.e. after 30 turns on average).

When a war ends, recent-embargo memory of all war parties is reduced to 1, meaning that embargoed war parties become willing to trade 9 turns after the war ends (on average).

Rationale	To make embargoes more useful and to help reconciliation after a war. Of course players don't want third parties to trade with their war enemies – this shouldn't upset the AI too much. Which is to say, being upset with a third party that agrees to the embargo does make sense.

Furious AI civs don't ask for embargoes. The AI also doesn't ask for an embargo if the relations value toward the human is the same as toward the worst enemy or only one higher.	Typically, if the AI is Furious toward a player, then that player is him-/herself the worst enemy. When the AI is Furious toward multiple teams, and the player is not the worst enemy, then the attitude toward the human doesn't prevent embargo requests.

Rationale	Not trading with other enemies of the AI is expected too much when already Furious. And shouldn't keep digging when relations are already at rock bottom.
	The relations-value clause should prevent worst-enemy oscillation.

If the AI attitude toward the (human) player that proposes an embargo is strictly greater than the threshold (i.e. Friendly in most cases), the attitude threshold for the embargo target is reduced by 1. However, except for Tokugawa, an AI leader still won't agree to stop trading with a civ that he or she is Pleased with.	The most common attitude thresholds require that the AI is at least Pleased with the human player and Annoyed or worse with the target of the embargo (else: "we couldn't betray our close friends"). Some AI leaders require Friendly or only Cautious attitude toward the human and are willing to stop trading at Cautious attitude toward the target. Only Tokugawa is willing to stop trading despite being Pleased with the target.

Rationale	The BtS thresholds for attitude toward the target are a bit strict – when Annoyed, there isn't going to be much trading anyway. This change also makes Friendly relations a bit more rewarding.

Regardless of attitude, the AI refuses to stop trading if this means canceling Open Borders with a dangerous civ: "We're afraid of their military might."	Military power isn't taken into account.
Rationale	Should make it harder to abuse the AI as a meat shield.
Power ratio is factored into the cost that the AI charges for an embargo.	Only the game turn, number of cities, attitude values and current deals matter.
Rationale	Even if a civ is not an immediate threat, an embargo makes a war at a later time more likely. Don't want humans to set a peaceful AI that is getting ahead in tech on a collision course with a militarily more powerful civ through an embargo; or at least not so easily.
The number of civs in the game factors into the embargo trade value.	
Rationale	An embargo is more consequential when there are few civs to trade with.
Tbd.	Just counting the civs alive is a bit simplistic. Could copy code from `IllWill::preEvaluate` for a start.
130g	Relations penalty for rejected demand lifted during war
When an AI civ has been at war for at least 10 turns, and it's a war the AI civ started, it forgets about tribute demands rejected before the war. Memory about demands is also erased when a civ signs a vassal agreement. The vassal forgets that its demands were rejected (but other civs continue to remember demands that the vassal rejected).	Rebuked demands are remembered for 150 turns on average regardless of war.
Rationale	To make reconciliation after war easier. (Also, tribute demands are perhaps more common with UWAI.) The interpretation is that the rebuke has been avenged through war. (If the war was unsuccessful, then the AI, apparently, had been wrong to make demands.) Erasing the rebuke memory directly upon declaring war could obscure the fact that the rebuke contributed to the DoW; the rebuke wouldn't be visible anymore on the Foreign Advisor screen. Hence the 10 turn delay.
130h	War-on-friend penalties
See also	130y deals with reduced declared-war-on-us penalties from vassals
When assigning war-on-friend penalties, master civs ignore their vassals, and vassal civs ignore their master and the master's other vassals. Bringing in a war ally does not lead to a penalty from vassals of the target (no change). No war-on-friend penalty from anyone for attacking a capitulated vassal.	When a master civ is attacked, the aggressor gets a diplo penalty for declaring war on the master and – if the master likes its vassals – a penalty for attacking the vassals. Similarly, the vassals resent the attack on their master, and each vassal may resent the attack on the other vassals (of the same master). Can get a penalty both for the master and its capitulated vassal.
Rationale	The "You declared war on us!" penalty suffices. Should make it a bit easier to have normal relations with a vassal after helping it break free. War on capitulated vassal: Relations toward a capitulated vassal should basically never matter; see change 130v.

No penalty if the AI civ has war-on-friend memory about the attacked civ and the attacked civ has recently attacked a liked civ (i.e. the liked civ has war plan "attacked" or "attacked recent").	
Rationale	If the AI is unhappy about a civ under attack, it shouldn't be unhappy about a third civ attacking the aggressor (even if the AI still likes the aggressor).
Tbd.	Not nice that this depends on information that the player can't see (AI war plan type). The following should help a little (but still ...):
Vassals are upset about nukes dropped on their master only if the vassal's true attitude is Pleased or greater.	The forced attitude counts (as in most situations).
Rationale	As above, to decrease war-related diplo penalties from enemy vassals.
The confirmation popup for a declaration of war lists all AI civs that are going to disapprove (if any). Triggered Defensive Pacts are also listed.	If war is declared through the diplo screen ("Your head would look good ...") or Alt+Click on the scoreboard, the "Does this mean war" popup appears. When attempting to enter closed borders, the "Entering will trigger war" popup appears along with "You can peacefully enter if you sign an Open Borders Treaty" if OB trading is possible.
Rationale	The list could be especially helpful in team games because the penalty is based on team-toward-team attitude and I'm not sure that these attitude values are shown anywhere on the UI.
Tbd.	Perhaps something similar could be done with the raze-city popup when the city is a holy one. Nukes also.

130i	Diplo modifier from Open Borders (OB)	
AdvCiv		*BtS*
Each turn that an AI civ has OB with another civ, the OB counter of the AI is incremented twice, each time with the same probability (cf. 130k). This probability is based on the AI civ's profit from trade routes with the other civ and the (spatial) closeness value (cf. 107) between the two civs.		The counter just counts the number of turns that the two civ have had OB.
Rationale	It had been too easy to please the AI, and, consequently, there wasn't enough warfare overall. See for example this game report of mine played with v0.87, prior to change 130i. In particular, civs that never had much to do with each other were often unwilling to go to war when they came in closer contact through Galleons or conquest.	
See also	149 has the same aim.	
Memory about OB decays at a rate of 1.45% when borders are no longer open.

The OB memory has an upper limit of 60. | | OB memory never decreases. When borders are closed, the relations bonus is suspended until they are open again.

The relations bonus is +1 for every 25 turns. Although the bonus is capped at 2, the OB memory keeps increasing after 50 turns with OB. |

Rationale	Mostly for consistency: all diplo memory and counters decay (130r, 130k). Regarding the upper limit: otherwise OB that have lasted for, say, 250 turns would never decay to the point of reducing the relations bonus.
130j	~~Friendly AI more resentful about bad actions, Annoyed AI happier about good actions~~ Changed scale of AI memory
Tbd.	The scale factor of 2 is currently hardcoded in several places and a higher factor (3) is used for declarations of war (DoW). Should either directly store the effect on relations in the memory counters (e.g. decrease the DoW counter by 300 when war is declared on the AI) or set a single scale factor in GlobalDefines. The latter approach would have the advantage that any changes that a player makes to the `AttitudePercent` values in XML would affect ongoing games and that there would be no danger of overflow. A scale factor of 10 should suffice, and then it would take e.g. 3000 nukes to cause a `short int` variable to overflow.

~~When an action displeases the AI, it adds 1 to corresponding memory if it's already Annoyed or Furious, 3 if it's Friendly and 2 otherwise. When an action pleases the AI, it adds 1 if already Friendly, 3 if Annoyed or worse and 2 otherwise.~~ ~~This results e.g. in -4 relations for a DoW on a Friendly civ, and only -2 if they're Annoyed.~~ (Disabled the attitude-based logic again.) Memory is increased by 2 in all cases.	The AI stores separate memory about each civ and for each type of action, e.g. DoW or an accepted tribute request. When a diplo action occurs, the respective memory is increased by 1. The current AI attitude has no impact on this. (Although the AI e.g. doesn't make tribute demands at Pleased.)
This faster increase of memory is evened out by 100% faster decay than in BtS for all memory types, and the impact of each remembered action is halved. Since each action normally corresponds to a memory value of 2, it's unlikely for an action to be completely forgotten after a few turns, or to be remembered fully for a long time. "You razed our cities/ a holy city" and "You nuked us" work differently; see 130q. (But not "You nuked our friend".)	The decay speed depends on the memory type and leader personality. E.g. Alexander remembers accepted tribute for 50 turns on average. That said, he could also forget about it within just 5 turns or still remember it after 115 turns (both ca. 10% probability).
Rationale	To make relations more dynamic. Also more plausible that a civ leader would be more indignant about a DoW by a friend than by an enemy. The decreased randomness is a welcome side effect. It doesn't make sense to base the diplo penalty for razing and nuking on attitude because these penalties only occur when already at war, and, then, the enemy attitude is usually Annoyed or worse. One could, alternatively, treat raze and nuke like the other diplo actions, but increase their base penalties; however, this would also affect razed cities with culture of non-war parties (which aren't necessarily Annoyed), and the BtS base penalty of -2.5 is already severe in this case. I've disabled the core of this change in v0.91 because it's a change that players need to know about, but that actually matters rarely. It can also create a counterintuitive incentive to keep relations below Friendly. The more fine-grained memory counting remains in place.
See also	130r uses an even finer scale for memory about declared war.
130k	Some randomness added to AI diplo counters
See also	130i deals with the OB counter, and 149 with resource trade.

Counters relevant for diplo increase and decrease probabilistically by either 0, 1 or 2 per turn.	For every civ, the AI keeps track of the number of turns spent in various relationships with that civ: war/peace, shared/opposed religion, shared civics, resource trade, defensive pact, OB, shared war. E.g. the defensive pact counter is increased by 1 on every turn that the two civs have a defensive pact, and decreased by 1 otherwise.
When the condition for increasing a counter is not met, the counter is decreased by 1.7%.	

Rationale	While memory decay (130j) is too unpredictable in BtS, the counters are too predictable. E.g. the "years of peace" bonus kicks in after exactly 60 turns.
	Exponential decay should make diplo bonuses from counters less sticky.

The probability of increasing or decreasing a counter is adjusted to the game speed setting. While an AI civ is in the Ancient era, the probability is divided by the Growth modifier (e.g. 150% on Epic speed); after the Classical era, the divisor is the Golden Age modifier (e.g. 125% on Epic speed); and, *in* the Classical era, it's the mean of Growth and Golden Age modifier (e.g. 137% on Epic).	No game speed adjustment.

Rationale	On slower speed, more trades and wars tend to happen throughout the game than on faster speed, but the difference is not that great, so the Golden Age modifier seems like a good compromise. The early game is special because early wars are rare, so the peace and Open Borders counters usually increase monotonically until the Iron Age or so, and at that point, the relations bonuses often make AI leaders unwilling to attack each other on Epic and Marathon speed. Hence the extra slow increase during the first game era.
See also	130r factors game speed into memory decay.

130l	Accepting an AI request reduces memory about a past rebuke and vice versa
AdvCiv	*BtS*
(disabled by default since v0.85)	
When an AI diplo request is approved by another civ, memory about previously denied requests of the same type is reduced by 1. Likewise, a denied request reduces memory about previously granted requests. Since each request normally adds 2 occurrences to memory (see 130j), subtracting 1 does not always have a visible effect.	Accepted and denied requests are remembered separately, and entirely so.
The following request types are affected by this change: help, tribute, change religion, change civics, join war, stop trading. Requests of differing types do not affect each other; e.g. granting tribute does not erase memory about denied help.	

Rationale	Another means to turn bad relations around, and vice versa. Fairly low-key, I think. Multiple AI requests in a row don't happen often, and only 1 memory is subtracted.
	After some testing, I find that I never pay attention to this, and the other mechanisms for more dynamic relations seem sufficient. Disabled in order to make the mod a little less complex.
Config	Can enable this change again through `ENABLE_130L` in `GlobalDefines_advc.xml`.

130m	Shared-war diplo bonus based on war success	
AdvCiv		*BtS*
+1 relations for currently sharing a war (no change), more than that only if the one side has had losses in the war (i.e. is at least in some need of assistance) and the other side has also had losses or has inflicted losses on the shared enemy (both count equally). The exact formula is complicated; see `AI_getShareWarAttitude` in `CvPlayerAI.cpp`. The total relations bonus is capped based on leader personality (no change). Moreover, the bonus can't go higher than 2 unless the teams have shared a war for at least 8 turns; another 8 turns are required for every further point.		The AI counts the turns spent together at war (shared-war counter), and for every 8 turns, increases relations by 1 up to a limit that depends on the AI leader. This counter never decreases, meaning that the relations bonus remains for the entire game. It is only suspended while the former war allies are at war with each other.
Success (and losses) in the shared war are remembered by the AI beyond the end of the war but decay by 1.5% per turn (even while the war still lasts). The relations bonus for a remembered shared war is reduced if the AI needs help in a current war but the former ally hasn't joined the war.		The shared-war counter also matters for Permanent Alliances. The AI only agrees to those after at least 40 turns of shared war.
The shared-war counter decays by 0.1 per turn (on average) when no war is being shared.		
Rationale	The everlasting relations bonus was reported as a potential bug for the Unofficial Patch 3.13 but didn't get patched, and I agree that the BtS behavior is probably deliberate. That said, a permanent relations bonus is clearly too big a reward for just staying at war. My change should ensure that you need to fight alongside the AI in (somewhat) good faith in order to get a relations bonus.	
	I don't think players need to know the specifics of the war success formula; I hope it works pretty intuitively.	
No shared war bonus if either war ally is a capitulated vassal.		Shared wars can bring a master and its capitulated vassals closer together.
No defensive pact bonus for capitulated vassals.		All vassal agreements are treated as defensive pacts when it comes to relations bonuses.
Rationale	Capitulated vassals are dragged into war by their masters. They should generally be unhappy about that. At best, the shared war experience can make up for this unhappiness.	

AI less likely to ask for military aid if the war is still recent; same frequency overall. If Annoyed toward the player, the AI asks for an embargo instead of joint war. Past wars by the human don't matter.	AI civs on whom a human civ has declared war in the past don't ask that human civ for military aid, but the current attitude isn't an obstacle.
Also less inclined to ask for war against a civ that the player recently made peace with.	Can't ask if there's peace treaty, i.e. for 10 turns; beyond that, recent peace doesn't matter.
Rationale	A joint war is asked too much from a civ that the AI considers to be an adversary. (Or they could ask, but they shouldn't be mad when denied.) Embargo requests should generally be more common when the AI is at war.
	If the player has recently been at war with the target, then, apparently, the player is unwilling or unable to defeat the target. Also don't want the AI to drag players into the same war repeatedly.
Fighting against units inside another civ's borders is extra effective at increasing the shared-war memory. Of course, these units need to be hostile to the tile owner. Losing a unit in such a fight counts as much as defeating a hostile unit.	War success has no impact on the shared-war diplo, no matter where it occurs.
Barbarians count as hostile in this context, i.e. it's possible to get a "mutual military struggle" bonus without even sharing a war, just by fighting Barbarians inside the other civ's borders. Fighting Barbarians elsewhere doesn't help though, and the bonus won't exceed +1 unless there is or was previously a shared war.	
Units with hidden identity (i.e. Privateers) also count (but not if the Privateer belongs to the plot owner).	
Rationale	If someone engages enemy units in the AI's territory, that someone is really being helpful, and not just competing for loot; the AI should appreciate that. Regarding Barbarians, I sometimes find that my, say, scouting Chariot could help an AI civ against Barbarian invaders, but BtS gives me no incentive to do so. The change should make these situations more interesting.
The relations breakdown says "Our mutual military struggle *has brought* us closer together" when not currently sharing a war.	Always present tense.
Rationale	"Our struggle brings us closer" sounds a bit strange when not currently struggling together. Might also help get across that the relations bonus is bigger while at war.
If UWAI is enabled, the AI does not ask the player to join a war when it thinks that this would be disastrous for the player. This is checked by letting the proxy AI (see 130u) compute the player's war utility. The AI may ask for an embargo instead.	The human position is not considered by the AI.
Rationale	Would prefer to let the player decide whether a joint war is a bad idea, but the player shouldn't be punished with a diplo penalty (help refused) for not being stupid.

130n	Adjust ideological relations modifiers (religion, civics) to popularity
See also	[148](#) statically (i.e. not dependent on the game state) reduces the limits of those modifiers across the board.
	[130x](#) makes these modifiers take a bit longer to attain.

AdvCiv		*BtS*
The different-religion penalty starts at -1 (or -2 with the Holy City) for all AI leaders. For the majority of the leaders, the penalty then becomes one worse after 15 turns and, for some of the leaders, another one worse after another 15 turns (well, 16 turns, see 130x). The total limits remain the same for all leaders.		The penalties start at 0 to -2 depending on AI personality (another -1 with the Holy City) and then grow one worse after 5 turns. In contrast, the same-religion and favorite-civic modifiers start at +1 for all leaders (+2 with the Holy City) and then increase by one every 10 turns.
Rationale	So that all ideological modifiers follow the same basic principle of starting at a small constant value and then growing up to a leader-specific limit. Before streamlining the AI behavior like this (this happened in AdvCiv 1.01), I had implemented separate but similar mechanism for adjusting positive and negative ideological modifiers.	
	Moreover, letting the different-religion penalty increase gradually gives civs that have founded their own religion in (semi-)isolation more time to make some tech trades with the rest of the world before the penalties make everyone annoyed. Letting the penalty grow more slowly than the positive modifiers also reflects that the penalty has (for most leaders) a tighter limit than the positive modifiers. I.e. the time that it takes to max out the penalty becomes more similar this way to the time that it takes to max out the positive modifiers.	
	A penalty of 0 for 5 turns doesn't make much sense to me. I guess it guarantees that civs can change their own religion before the penalty applies, but I don't think that's something players will commonly do, and having a penalty of 1 apply for a few turns is hardly a problem.	
The limits for the time-based ideological modifiers are decreased based on the portion of known cities whose owner follows the ideology in question. I.e. all revealed cities are counted so long as their owner runs the proper state religion (or civic), regardless of the religions present in the cities. For the civ toward whom the modifier is computed, even unrevealed cities are counted.		No such dynamic adjustment.

	The high diplo bonuses from early civics, especially Hereditary Rule, are too easy to get, and BtS diplomacy gets boring when everyone's in the same religion. Also to reduce the overall (irrational) impact of ideologies on AI relations, considering that change 148 reduces the threshold for Friendly relations (effectively) by 2 and that change 130n reduces the limits for positive ideological modifiers only by 0.5. The change seems also realistic: if everyone shares a trait, it gets taken for granted, and lack of external enemies leads to infighting. And it's unwise to eschew everyone when they all have "heathen" religions. It's not so clear how the popularity of an ideology should be measured. Counting cities rather than population makes it less important which tiles, exactly, are revealed. I don't want players to worry about hiding or revealing their own cities or those of third parties. Hence also the exception for the civ toward whom AI attitude is being computed. This small vision cheat seems unlikely to leak exact city counts to observant human players, and a rough estimate can be guessed from the player score anyway. An important upside of disregarding unrevealed cities of third parties is that, this way, the adjustment is mostly based on those civs that actually interact with each other. E.g. when everyone on one continent is in the same religion, the same-religion modifiers should be reduced a lot (i.e. about halved) at least until the other continents (or their shores) have been explored a bit.
The different-religion penalty is also reduced when the offending religion has few adherents (again, measured by revealed cities).
Rationale	To make civs whose religion has failed to spread less unpopular. Also makes sense to me that a religion with few followers would be more easily tolerated; it's not threatening. Note that the adjustments to the different-religion penalty usually just reduce the penalty by one if at all because the time-based portion is only between 0 and -2.
130o	Changes to made-demand memory
See also	130v prevents vassals from acceding to tribute demands from rivals. 144: refusal of gift request.

When a human player declares war on an AI civ (primary DoW; not through DP), and that AI civ remembers having paid tribute to the player, the AI civ and all non-vassal AI civs that it has met set their recent-demand memory about the player to 8, which (due to change [130j](#)) means that they won't consider any requests for help or demands for tribute by the player for (on average, on Normal speed) 80 turns. The declare-war confirmation popup warns the human player about recent-demand memory	Accepting tribute results in a 10-turn peace treaty. After that, there is no particular penalty for declaring war despite tribute.
If the AI denies a demand for tribute, the human player does not receive a diplo penalty. Only granted tribute results in "You made an arrogant demand", and this memory decays after 30 turns on average. I.e. the only negative consequence of a denied demand is that the recent-demand memory increases. Accepting tribute still results in a peace treaty. When an AI civ declares war (primary DoW) or signs a vassal agreement, it sets all its arrogant-demand memory to 0.	"You made an arrogant demand" regardless of whether tribute was granted; never goes away.

Rationale	The BtS mechanism incentivizes players to demand tribute from civs about 10 turns before attacking them anyway. Not at all how tribute should intuitively work, not how it works when the AI asks tribute from a human player, and not an interesting decision. Could go about improving this in many ways. My approach is supposed to be low-key and low-effort. I've removed the penalty for failed demands because players need to be able to tell if a civ has actually paid; if it hasn't, it's OK to attack. Moreover, a penalty for a failed demand encourages reloading. Not sure if the penalty (no requests for 80 turns) is painful, but at least it makes it impossible to routinely receive tribute before going to war. Not ideal that the safety period is tied to memory decay, and thus randomized. Wouldn't make sense to apply this change to AI civs that declare war after receiving tribute from humans; AI civs never request or demand anything from each other, so increasing AI-to-AI recent-demand memory would have no effect. An earlier implementation kept the "arrogant demand" penalty for unsuccessful tribute demands, and added a new type of diplo memory ("exacted tribute") to distinguish the two outcomes.
Config	The 80 turns are customizable in `GlobalDefines_advc.xml`.

Tbd.	Arrogant demand should be remembered for a non-randomized period of time. Also, 30 turns is too long; try 25.
	Would be nice to show info/ a reminder about this change on the diplo screen, before or after the player makes a demand; however, the text with which the AI responds to a demand is also used for vassal tribute (which doesn't cause "arrogant demand" memory) and "We demand that you give us this in tribute" mustn't take up more than one line because it's always shown while the player arranges trades. Could show a message I guess (outside the diplo screen) ...
	Would like rejected gift requests to add only 1 to recent-demand memory; lower stakes that way. Would have to move code from `CvPlayer::handleDiploEvent` into `CvPlayerAI::AI_considerOffer` for this.
See also	[130h](#) also adds warnings to the DoW confirmation dialog.

The AI remembers granted tribute demands for, on average, 30 turns. Rebuked demands are remembered for 120 turns.	50 turns/ 150 turns
Rationale	To match the arrogant demand memory. And the rebuke memory decay was just too slow generally in my opinion.
Config	Implemented by reducing the BtS values read from XML to 60% in the DLL. Because I didn't want to change the values for every single leader in XML.

Gandhi gets as upset as most other AI leaders (i.e. -1 relations) when he pays tribute.	Gandhi is the only leader who doesn't mind tribute demands.
Rationale	Otherwise, the new mechanism couldn't apply to Gandhi, which would be strange: why do the other AI civs not care when Gandhi is attacked after paying tribute? Gandhi has still enough other quirks; he may well be the leader with the most extreme personality overall.

When the AI disregards a request for a gift or a tribute demand because it still remembers a recent request or demand (both based on recent-demand memory), recent-demand memory is only increased by one, meaning it'll take half as long as in BtS until the next request/ demand is considered. Making requests and demands can't increase the recent-demand memory beyond 2.	Each request adds fully to the recent-request memory, meaning that, if the AI still remembers one recent request when a new request is made, it takes 40 turns on average until another request is considered. Also, each tribute demand (regardless of success) further worsens relations, up to a maximum of 10 demands. I.e. one can worsen relations almost arbitrarily within a single turn.
Rationale	Thanks to change 130j, there's now a lighter punishment available for aggressive begging. 40 turns is awfully long.

If UWAI is enabled: The AI remembers for 10 turns on average (plus the duration of the peace treaty) whether a human has recently accepted a demand. While the peace treaty from the demand lasts, the AI doesn't plan war against the human. After the peace treaty, so long as the demand is remembered as recent, the AI treats the human as if its attitude was at least Pleased (as far as war planning is concerned).	Only remembers "You gave us tribute" (for 50 turns on average) but not whether the tribute was granted recently. The BtS AI never plans war during a peace treaty, but the K-Mod AI does; and the K-Mod AI is able to declare war shortly after the peace treaty ends.

Rationale	Since humans are now encouraged not to attack shortly after receiving tribute, it seems like a double standard when the AI does it.

130p	Fair-trade bonus and enemy trade penalty reworked – and some vaguely related changes to cancellation memory about OB, DP.
Rationale	Hopefully, in the end, the bonuses work as one would intuitively expect. The details have gotten pretty complicated (and especially complicated to describe).

AdvCiv	*BtS*
Each time a trade is made, a score based on the normalized gold value of the traded items and the current asset totals of both parties is recorded. The accumulated score decays by 1.45% each turn (Normal speed). The relations modifier computed from the remembered score is subject to diminishing returns, meaning that more trading is needed in order to get from e.g. +3 "fair and forthright" trade to +4 than from 0 to +1. The modifier is increased a bit if the two civs have only recently first met.	Raw gold values are recorded without adjustment for game speed or game progress, and don't decay. When computing relations, the total recorded gold value is divided by the has-met counter, i.e. the number of turns that the two civs have known each other.

See also	[131](#) makes some minor adjustments to asset totals.
Rationale	The BtS formula works OK for Normal speed in the first third of the game but becomes too sensitive later in the game (because trade values increase faster than the has-met counter), and on slower game speed settings. It also produces weird results for civs met in the middle of the game; a small gift can be enough for a +4 bonus or -4 penalty, and modifiers can decrease rapidly as the has-met counter increases. I'm taking a more explicit approach by recording normalized trade scores and applying exponential decay each turn (instead of implicit decay through increasing trade values and has-met counters). Asset totals seem well suited for computing trade scores because they tend to increase at a similar pace. Despite the decay, trade scores still tend to be higher between civs that have known each other for a long time than between civs that have recently met. It does, therefore, make sense to take into account the has-met counter, but the effect is now much smaller than in BtS.
Tbd.	Make this more sensitive when tech trading isn't allowed.

The fair-trade relations modifier is multiplied by `MemoryAttitudePercent`-`EVENT_GOOD_TO_US` in `Civ4LeaderHead.xml`, which, however, is 100 for all leaders, so this matters only if a mod-mod changes some `MemoryAttitude` values. For enemy trade, `MemoryAttitudePercent`-`EVENT_BAD_TO_US` is used instead. This is also 100 for all leaders.	Not based on personality at all, neither through a multiplier, nor through bounds. (The bounds are hardcoded as -4 for enemy trade and +4 for fair trade; this is still the case in AdvCiv.)
The fair-trade modifier is based on the difference between the recorded values of items that the AI has received in trade minus 4/5 of the recorded values of items that the other side has received.	100% of what the other side received is subtracted. I.e. a human who makes only deals with the AI that the AI finds perfectly fair never receives a "fair and forthright" relations bonus.
Rationale	It's clear enough that "fair and forthright" is supposed to be a bit of a euphemism, but

trades that are actually fair should still have some impact. This should also make it a bit easier for AI civs to reach mutually Friendly relations (despite change 148, AI-AI friendships are a bit scarce). The coefficient was 5/6 instead of 4/5 until AdvCiv 1.04.	
Open Borders (OB) with a worst enemy contribute significantly to the relations penalty. The impact is based on the number of turns that the two civs have had OB. None if the borders aren't currently open.	OB and DP factor into the trade memory for both fair trade and enemy trade, but the deal value is so small that the impact is negligible.
OB (and Defensive Pact; DP) have no impact on the fair-trade bonus, and don't contribute to trade memory. Resource trades add to trade memory, though the impact is normally small. The trade value for the first ten turns is processed when the deal is signed, then, once the deal can be canceled, trade value is added turn by turn.	Resource trades don't matter for trade memory. All ongoing deals can cause the AI to demand a trade embargo.
The AI refuses to sign a DP if a DP was recently canceled (explicitly canceled or through a war declared by one of the signatories).	The AI refuses to sign recently canceled OB, but, for DP, there is no such restriction.
Rationale	OB can be very helpful for the enemy civ, may even be crucial at wartime, so I don't think the potential -1 from refusing a trade embargo is a sufficient penalty. I prefer to keep OB and DP out of the trade memory, so that cancelation of those deals immediately reduces the enemy trade penalty. This can't be abused by suspending OB and DP just for one turn because of the AI's refusal to sign OB and DP if recently canceled. DP needed this kind of restriction anyway; too easy to flick DP on and off in BtS.
See also	130t factors attitude and worst enemy into anger about rival DP. kekm.3 keeps DP intact after a foreign DoW. 550a gives civs that have fallen behind better deals by adjusting trade values. This adjustment factors half into the trade score relevant for relations.
Tbd.	May have to dial this up once there is tech diffusion from OB. Peace treaties should perhaps also add to the enemy trade penalty. Can currently ask for a gift or tribute to avoid getting asked to join a war.
If the AI itself cancels OB or DP, then it forgets the cancellation twice as fast (5 turns on average).	Doesn't matter who canceled: canceled OB are forgotten after 10 turns on average.
Rationale	It's frustrating when the AI attitude dips below the OB threshold (which is difficult to notice in time) and, on top of having to get the attitude back up despite having lost the OB diplo modifier, the human player has to wait for 10 turns (and often longer than that). It does make sense that the AI is a little wary about signing the agreement again – won't be able to cancel it for 10 turns then, and the other side has just proved to be unreliable (at keeping the AI happy). It also makes sense that the AI is, on top of that, upset when *the other side* cancels an agreement, possibly (the AI can't check this …) at an especially inopportune moment. Important that humans can't just briefly close their borders when it suits them (e.g. when an AI settler or army wants to move through); that makes the AI look dumb.
Tbd.	10 turns generally seems pretty long, also between two AI civs: both memory counters need to reach 0 and one side's contact roll has to succeed before a new deal can be

	implemented. Reduce to 8 turns on average?
"Our defensive pact proves ..." applies only when currently in a DP.	Doesn't check if currently in a DP. The relations bonus is based on the number of turns spent in a DP. That number counts backwards when not in a DP.
Rationale	More intuitive this way, and consistent with how the OB relations bonus works.
Reduced impact of OB and resource trades on the enemy trade penalty if the worst enemy has OB with many civs.	
Rationale	An AI civ can't afford to be mad at everyone. This is less of a problem for tech trades because one civ can only do so much tech trading.
Decreased the impact of gifts vs. trades on enemy trade memory a little.	The AI tracks gifts and traded items separately. Gifts are hated three times worse than traded items.
Rationale	I guess the idea is that trades aren't so bad because the worst enemy has to give something away too. But trading is dominated by tech trades, and giving away tech doesn't really hurt the worst enemy; it's very much win-win.
Reduced impact on enemy trade if the AI relations toward the enemy and the enemy's trade partner are similarly bad. When the worst enemy of an AI civ changes, that civ reduces its enemy trade memory about trades with its (former) worst enemy by one third.	When an AI civ dislikes two civs equally, and one of these enemies trades with the other, the distinction of "worst enemy" can oscillate between the two. Even if the AI reconciles with its worst enemy, it continues to be mad at civs that previously traded with its former enemy.
Rationale	Both changes are aimed at preventing situations where a civ that trades with the worst enemy becomes itself the worst enemy. This can seem erratic. (That said, even when the worst enemy changes, "you've trade**d** with our worst enemies" remains true.) I've considered halving the memory, but that could drop a -3 penalty to -1 due to rounding, which seems a bit much.
See also	[130f](#) prevents stop-trading demands when relations toward the player are already almost as bad as toward the worst enemy.
When picking the worst enemy, civs with whom the AI is at peace are only considered when the AI isn't at war with anyone. Exception: Dogpile wars (e.g. on request of another civ) have no impact on worst enemy.	Worst enemy chosen only based on attitude.
Enemy trade memory is increased more when the AI is at war with the worst enemy than when they're at peace.	Whether the worst enemy is a war enemy doesn't affect enemy trade memory.
Trades with a war enemy increase enemy trade memory even if that war enemy is not the worst enemy, though not as much as trades with the worst enemy do.	Only trades with the worst enemy count as enemy trade.
Rationale	Trades with a war enemy are more likely to harm the AI than trades with some cold-war foe.
See also	[552](#) increases the AI trade value of military tech. [148](#) changes the relations threshold for becoming worst enemies. [130d](#) changes some rules regarding vassal agreements and worst enemies.

	The Dawn of Civilization mod takes geographical distance into account when choosing worst enemies. Git commit (not always properly updating cached worst enemies I think)
Payments in peace deals (between two parties or brokered by a third party) increase enemy trade memory (but not fair-trade memory). Exception: Civs don't count enemy trade memory if they like the civ with whom peace has been made (Pleased attitude). Payments in two-party peace deals don't count *fully* for enemy trade memory.	Peace deals don't count for trade memory.
Rationale	Don't want players to bypass enemy trade penalties by piggybacking regular trades on (brokered) peace deals. It makes sense to me that a civ gets upset when its worst enemy is paid for peace.

130q	Nuke and raze memory based on city size
See also	650 deals with other changes to nuclear war (minor so far). 130j (partly disabled) exempted raze and nuke memory from being affected by attitude. advc.ctr bases liberation memory on the trade value of the liberated city.

AdvCiv	*BtS*
When a city is nuked, the AI owner remembers this as 1 to 3 bad actions depending on the importance of the city. Nukes that don't affect a city, including intercepted nukes, are counted as 1. If the nuke is counted as 1, partners of the nuked civ don't get upset ("you nuked our friends"). Razed cities count as 2 bad actions in the case of a significant city, otherwise 1.	Each exploded nuke counts as 1 bad action (which is equivalent to 2 bad actions given change 130j), even if it only hits a ship in the middle of the ocean, and nukes always trigger "you nuked our friends". No diplo penalty for intercepted nukes. Each razed non-holy city counts equally (unless it has 0 city culture and a dead civ has the highest city culture; see change 099).
Rationale	It should make a difference whether a city is hit at all, and whether it's, say, the capital, or some backwater. And a failed attempt to nuke a player shouldn't go entirely unpunished. A comment in the code suggests that Firaxis saw that as an open issue: "*Intercepted!!! (XXX need special event for this...)*" Well, it's a little unclear ... Raze anger generally seems a bit high, or just ineffective at discouraging razing. For now, lower the anger a bit overall by exempting minor cities.
Tbd.	The raze mechanism needs to change so that it takes multiple turns (or multiple units). The raze popup should say how great the anger will be. Hover text while firing a nuke ("Nuke Mode") should say which civs are going to be upset. An intercepted nuke that wouldn't have hit a city perhaps shouldn't add any nuked-us memory. A bit awkward to implement though.

Razing results in 0 memory only if a city has neither produced any culture nor population. (That means, 0 memory is only possible when auto-razing – but the converse is not true.)	0 memory for razing cities with 0 culture – no matter the size.
Rationale	A city can develop for quite some time, growing population, constructing a Granary, without producing any culture, and the AI should not be indifferent about losing such a city. I still want 0 raze memory for cities that have really just been founded (not much different from attacking a Settler) or that have been founded in awful terrain (can't blame the new owner for razing that).
See also	Kek-Mod marks cities that will auto-raze on the map (Git [commit]). While my change makes it a little bit more important to know whether a city will auto-raze, I still don't find it important enough.

The AI isn't upset about a nuked friend if the owner of the nuke is or used to be a friend that has itself been nuked at least as badly.	The AI only checks whether it's at least Cautious toward the victim.
Rationale	Don't hate both sides of a nuclear war if it's just tit for tat.

AdvCiv	*K-Mod*
AI doesn't disband nukes when in financial trouble unless no units other than cargo units remain.	As far as I can tell, nukes are disbanded before most other units. BtS tried to save expensive units, which is fine in the case of nukes, but K-Mod focuses on XP, which is generally better, but fails for nukes.

130r	All AI diplo memory decays; see the [table] in the <u>Dynamic Diplomacy</u> chapter.
See also	[130y] counts less declared-war memory in certain situations. [130o] deals with memory about tribute demands.

AdvCiv	*BtS*
The more incidents of a kind the AI remembers, the faster it tends to forget each of them. For example, a single declaration of war is, in the expected case, completely forgotten 150 turns after it ends; two declarations after 75+150 turns, three after 50+75+150 etc. Tech trade memory ("shared discoveries", "fear you're becoming too advanced") is exempt from this change.	The probability of decreasing the momery count does not depend on the current memory count. So three
Rationale	At least before the late game, there should generally be a route to reconciliation (though it doesn't always have to be worth following).
See also	[553]: Changes to the "shared discoveries" modifier

All decay probabilities and contact delays are adjusted to game speed using the same (moderate) modifier as for Golden Ages: 100% slower decay on Marathon, 25% slower on Epic, 20% faster on Quick. No speed-adjustment for contact probabilities (as in BtS).	The various AI memory values are decremented each turn, each with its own probability from `Civ4LeaderHeadInfos.xml`. The probabilities are the unaffected by the game speed setting. After contacting a human player, the contact delay prevents the AI from contacting the same human player with the same type of request. Not speed-adjusted.

Rationale	Help requests – a pretty common type of AI request – should not really be made more frequently on slower speed settings; they should match the research speed. Other requests, e.g. for joining a war or enacting an embargo, should match the pace of diplomacy, which is not nearly as much affected by game speed. The Golden Age modifier seems like a reasonable compromise. I don't want to adjust the contact probabilities because that would make the AI slow to respond to game state changes e.g. someone starting to trade with an enemy.
	Decay speed: If the contact frequency is reduced, then individual requests should arguably be remembered longer so that the typical number of active diplo modifiers is unaffected by game speed; at least, there shouldn't be *fewer* active modifiers on slower game speed. And a granted help request shouldn't be forgotten as fast on slower game speed settings because the gifted technology took longer to research.
	Another (minor?) reason for slowing down the decay rates: I don't think the AI really contacts human players three times more often on marathon than on Normal speed in BtS. Apart from a successful dice roll and expired delay, certain other conditions need to be true for each specific request. For example, an AI civ can only make so many help requests in a row until it becomes too advanced to receive further help.
	On the other hand, an argument against slower decay on slower speed: Some diplo memory, e.g. "you declared war on us", isn't affected by contact delays. Still, I think declarations of war don't happen as often (in terms of game turns) on slower speed as on Normal speed, so (moderately) slower decay shouldn't lead to problems with penalties stacking up too much.

See also	130k factors game speed into state counter (e.g. "years of peace") increments. 130p: Remembered trades ("fair and forthright", enemy trade) don't use the MemoryDecay system. The remembered trade values match the game speed because e.g. techs have higher cost and thus higher trade value on slower game speed settings. CFC post of mine about game speed adjustment of diplo parameters; and another post (after the 1st quote box). Discussion specifically about Open Borders contact delay.

Transformed the independence bonus into a memory-based bonus to let it decay. It's treated as 10 memory decaying at a rate of -1 every 30 turns on average (Normal speed).	A liberated colony is forever +10 grateful to its old homeland.

Rationale	To be consistent with the "everything decays" paradigm.

War success decays by 3% each turn; that's a reduction to 75% after 10 turns. It's still set to 0 at the end of a war.	War success doesn't decay; only reset to 0 when a war ends.

Rationale	In long wars, initial successes tend to have too much weight.

Tbd.	Initial successes should perhaps be ignored entirely, or decay very quickly because they're often owed to surprise and thus not a good estimate for the future course of the war. Resetting war success at the end of a war isn't smart because war could break out again just 10 turns later, but 0 war success while at peace might be assumed by some of the code, so I'm not sure how easy it would be to change this.
See also	<u>UWAI</u> remembers the overall outcome of a war before war success is reset.

No decay of declared-war memory while the war lasts. Same goes for war-on-friend memory while at war with a partner.	No decay of either memory type ever.

Rationale	Plausible that forgiveness can't happen while the transgression is ongoing. Also, war-on-us memory was decaying too fast without this restriction.

Recently-canceled memory (about OB, DP and VVA) can only decay half (from 2 down to 1) while at war.	No recently-canceled memory about DP and VVA. AI often willing to sign OB right after a war ends.

See also	<u>130p</u> introduces recently-canceled memory for DP and <u>143</u> for VVA. <u>130j</u> counts memory at a finer granularity, which is why canceling a deal sets recently-canceled memory to 2.
Rationale	AI shouldn't generally be willing to sign agreements right after a war ends.

"You stopped trading with us" (`MEMORY_STOPPED_TRADING`) is remembered for 60 turns on average, same as "You negotiated a trade embargo," and neither of these decay while there is recent stopped-trading memory (`MEMORY_STOPPED_TRADING_RECENT`).	When a team agrees to an embargo, the invisible `MEMORY_STOPPED_TRADING_RECENT` count is incremented and lasts for an average 25 turns. Additionally, the `MEMORY_STOPPED_TRADING` count is permanently increased, leading to the relations penalty.

Rationale	The relations penalty shouldn't disappear while the embargo is still in effect.
See also	<u>130f</u> makes `MEMORY_STOPPED_TRADING_RECENT` decay a bit faster than in BtS.

130s	Accepting to join a war gives +1 relations	
AdvCiv		*BtS*
When a human player accepts an AI request for war aid, this is remembered as "You agreed to come to our aid in wartime." for 100 turns on average (on Normal speed). The bonus is suspended when the human civ isn't sharing any war with the AI civ and the AI civ is fighting at least one war.		The relations effect, including the explanation text, is implemented, but disabled in `Civ4LeaderHeadInfos.xml`. The duration is set to 150 turns on average.
Rationale	The thankless help requests bother some players; military assistance shouldn't be treated as a matter of course – that's what defensive pacts are for. Perhaps the original developers felt that the shared-war bonus is enough reward for a declaration of war, but now that the shared-war bonus is harder to get (change <u>130m</u>), it should be OK to reward the granted request directly. 150 turns seems a bit much though.	
Config	Can be disabled in `GlobalDefines_advc.xml`. (Or in `Civ4LeaderHeadInfos.xml`, but there, the change would have to be made for each leader individually.)	

See also	[130m](#) should make the timing of help requests more predictable.
	[104i](#) (UWAI) makes the target of a joint war refuse to talk as in BtS, thus preventing the hired civ from making peace immediately.

Accepting a join-war requests from Gandhi results in the same relations boost as from the other AI leaders.	No relations penalty for denying a join-war request by Gandhi.
Rationale	Gandhi specialty is that he doesn't resent denied requests – which may fit well enough with his ideas about self-sufficiency. If there were also no consequences for accepting his requests, then they might as well be disabled. But I see no problem with him being grateful for help, even military aid.
Tbd.	The contact frequency is a different question. Gandhi generally asks for handouts too often.

130t	Diplo penalty for Defensive Pact based on relations toward third party
See also	[130p](#) makes the AI refuse a DP if recently canceled, and exempts DP from enemy trade and fair trade diplo.

AdvCiv	*BtS*
The relations penalty from an AI civ X toward a civ Y for having a defensive pact with a rival Z of X is based on the attitude of X toward Z. There is no penalty if the attitude is one higher than the `DeclareWarThem` threshold, or when X also has a DP with Z, or when X is too weak to attack Z regardless of the DP.	No penalty if X has a DP with Y; attitude doesn't matter.
The `DeclareWarThem` threshold is at Annoyed for e.g. Darius, at Cautious for e.g. Hannibal and at Pleased for e.g. Julius Caesar. (And at Friendly for Catherine, but this is no different from Pleased in this case because I'm using the threshold plus 1, and Friendly is the highest possible attitude.)	
Rationale	BtS doesn't properly factor DP into enemy trade diplo, and I think it's better to have all the negative DP diplo in one place anyway. A DP of Y with both X and Z shouldn't satisfy X and Z unless they like each other. In this constellation, Y threatens war on X if X should attack Z, which should offend X. Y also promises to protect X; but this is covered by the DP relations bonus (no change to that).
	As for the threshold, I've considered using the `DefensivePactRefuse` threshold from `Civ4LeaderheadInfos.xml`, which is Cautious for some leaders, but a leader that is quick to sign a DP shouldn't necessarily be forgiving about a rival DP. (The opposite could be true: these leaders think that DPs are important, and are therefore worried about rival DPs.)
	Warmongers tend to have high `DeclareWarThem` thresholds (easy to convince to declare war). These leaders should be bothered most by DPs.

AI refuses to sign a DP without OB: "Surely, you must be joking."	
Rationale	How are you going to defend us without entering our borders??

AI refuses to sign DP when no rivals remain.	Refuses when no other teams remain.

Rationale	Probably an oversight; should've been changed when vassal agreements were introduced in Warlords.
Credits	crullerdonut reported the issue in this CFC post.
Anger about rival peace vassals works the same way as anger about defensive pacts.	see 130w
Rationale	Voluntary vassal agreements are much like defensive pacts.
See also	About capitulated vassals, see 130w.

130u	Proxy AI attitude	
AdvCiv		BtS
The AI running in the background, ready to take over for a human player, is Friendly toward members of its team, Annoyed toward its human rivals and either Cautious or Annoyed toward its AI rivals depending on the AI attitude toward the human.		The proxy AI computes its attitude just as if it were in control. K-Mod 1.46 instead makes the proxy AI Cautious toward everyone. The proxy AI attitude factor into various AI calculations that determine how threatening a rival player is.
Rationale	The true attitude of the proxy AI isn't a good predictor of human behavior; in particular, the AI should never assume that a human rival won't declare war.	
Tbd.	Before setting the human attitude in `CvPlayerAI::AI_getAttitudeVal`, I added custom code in a bunch of places for setting the attitude that the AI should assume for a human player. Most of that code is probably no better than the newer code in `AI_getAttitudeVal`, i.e. at best redundant, and should therefore be deleted.	
See also	130v lets capitulated vassals adopt their masters' attitude toward other civs.	

130v	Masters are held responsible for their cap. vassals; cap. vassals as zombies
See also	K-Mod disables vassal-master attitude sharing; UWAI disregards capitulated vassals as war targets; 130d makes capitulated vassals ineligible as worst enemies. 130t bases the "vassals to your empire" penalty on attitude in the case of peace vassals, and 130w penalizes expansionism through vassals and conquests. 130y and 130h reduce war-based diplo penalties for vassals, recognizing that vassals don't have a choice in starting wars. 099c prevents master cities from flipping to a vassal, and 025 reduces the culture spread of capitulated vassals. 143b prevents capitulated vassals from building nukes. 014 makes capitulated vassals ineligible for team votes, stops them from pursuing victory and from building great wonders. kekm.25 prevents capitulated vassals from defying resolutions. 112b changes conditions for capitulation. 130f lets capitulated vassals agree to embargoes for free. 036 (gold available for trade based on attitude) disregards the attitude of capitulated vassals. 033 stops capitulated vassals from training Privateers. CFC discussion about the relationship between capitulated vassals and their masters: link
Tbd.	I'm considering to turn capitulated vassals into voluntary vassals after some time; perhaps after 30-40 turns. Would only need to implement a popup asking the master to sign the new agreement or else free the vassal. A capitulated vassal that reaches

the territory and population threshold should then also into a voluntary vassal.	
Some way for a master to free a voluntary vassal would also be nice.	
Capitulated vassals should perhaps only be allowed to trade tech with their masters. Or just prevent them from brokering to other civs. That said, the AI currently keeps its vassals (capitulated ones too) at arm's length tech-wise; that might have to be changed if vassals can't trade or broker.	

AdvCiv	*BtS*
If a capitulated vassal • razes a (holy) city; • trades with someone or someone's enemy; • has OB with someone's enemy; or • nukes someone (but not just someone's friend) half of the resulting relations modifier is applied to the vassal, and half to the master. Relations penalties about shared borders are also shared between vassal and master. In team games, the penalties apply to the leader of the master team.	Actions of a vassal don't reflect on the master, but civs are generally upset about "rivals" being vassals to the master's "empire", and vassals are possible war targets. Prior to K-Mod, the AI attitude toward the master was generally lowered based on the attitude toward the vassal.

Rationale	My thinking is that there shouldn't be grudges against capitulated vassals because they're basically dead already. This opens up a loophole: A human master could let vassals do the dirty work (razing), or use them as buffers against shared-border anger. BtS attitude averaging would fix this issue, but would also punish the master for e.g. the vassal's religion and any bad deeds prior to the vassal agreement. The change about nukes is obsolete because 143b now prevents capitulated vassals from having nukes.

Vassals are always Friendly toward their master, and capitulated vassals share the master's attitude toward rivals, but their attitude can be at best Cautious. AI civs project their attitude toward a civ onto the capitulated vassals of that civ. This does not apply when a vassal is deciding whether to break free. Masters are Pleased toward their capitulated vassals, and capitulated vassals are Pleased toward fellow vassals with the same master.	Voluntary vassals are Friendly toward their master except when deciding whether to break free. The attitude of and toward capitulated vassals is computed normally but rarely matters. Sometimes a (capitulated) vassal has a much more positive attitude toward another civ than its master, and will trade away technologies that the master wouldn't trade.
Vassals never grant gifts to rivals and never ask rivals for gifts. When deciding whether to grant a gift to its master (no matter if capitulated), the vassal checks its actual attitude (what the attitude would be if it weren't for the vassal treaty). Capitulated vassals are slightly reluctant to found cities, especially on other continents. AI Spies are reluctant to attack a rival's	When a vassal grants a gift to a rival or receives one, a peace treaty is signed between the master and the rival. Thus a civ with multiple vassals can sometimes be kept at peace indefinitely. Capitulated vassals with few cities left tend to found cities in marginal spots because the additional expenses (esp. for number of cities and civics) are small. The additional maintenance for the master isn't taken into account. AI doesn't attack it's own vassals (still the case in

capitulated vassals.	AdvCiv).
Rationale	Given my other changes, the attitude of and toward capitulated vassals really doesn't matter anymore, and showing them, as is often the case, Annoyed toward their master is misleading. I'm still showing the actual relations value (e.g. "Friendly (-2)") and modifiers because they do matter again if a vassal breaks free.
	If vassals are supposed to be zombies, they shouldn't be relevant for tech trading, and they generally shouldn't undermine the master's foreign policy. Until v0.85, I had capitulated vassals share the master's attitude toward other civs, but this made it too easy for a human player to dictate civics and siphon gold from vassals with AI masters; now they're at best Cautious toward rivals.
Civs that can't declare wars can't sign peace treaties at all. Their wars end without a peace treaty. This applies to both types of vassals and when a game option prevents war. Exception: A passed force-peace diplo vote will still result in peace treaties for vassals.	Apart from tribute and gift requests, vassals sign a peace treaty when their master ends a war (I think). Perhaps also in other situations that I can't think of.
Rationale	It's enough for the master to have a peace treaty. If a vassal agreements ends within 10 turns of that – fair enough, the vassal shouldn't be bound by the master's treaties anymore (this argument doesn't apply to diplo votes). This change also makes sure that a master can't somehow be forced into a peace treaty via its vassal.
Added a trade denial reason "We don't truly like you that much" for cases in which the true attitude is decisive for trade denial, doesn't pass the denial threshold whereas the forced attitude would pass it.	
See also	Only used for ctr so far; and I'm not sure it'll ever matter there.
Capitulated vassals are not shown on the "Glance" tab of the Foreign Advisor.	The Glance tab shows relations between all living (non-minor) civs.
Rationale	The Glance tab gets quite crowded in large games, and capitulated vassals don't have interesting relations because they follow their masters in matters of diplomacy.
Human espionage weight against vassal set to 0 after accepting capitulation.	No automatic changes to human espionage weights ever.
Rationale	Easy to forget, and cases in which a player would want to continue espionage against his/her capitulated vassal should be extremely rare. I'm not making the same change for voluntary vassals because these agreements are often fleeting.
AdvCiv	*K-Mod*
Capitulated vassals support all resolutions proposed by their master.	Team members of the Secretary-General (SG) support all resolutions. Civs Friendly toward the SG support some proposals, but not all (non-proliferation, force-civics).
Rationale	Since all votes are cast simultaneously, vassals can't always vote along with their master – they don't know how the master is going to vote.
See also	kekm.25 allows vassals to support the master in repealing resolutions (which requires them to vote "No" on the master's proposal).

130w	Penalty for expansionism: "We oppose your ruthless expansionism"
AdvCiv	*BtS*

Up to -4 based on the number of owned cities with foreign majority culture (depending on the difference between highest culture and owner's culture), the total number of cities, and the personality of the AI leader (peace weight minus warmonger respect). The penalty is reduced if the AI civ's military can't compete with that of the expansionist civ. The AI is (much) more sensitive about cities it previously owned. Other than that, so long as just about a quarter of a civ's cities are foreign, none of the other civs will mind. Up to -5 for the number of capitulated vassals, typically about -1 per vassal, but it depends on the number of cities owned by the vassal, the initial number of civs, and (as above) AI personality and power ratio. This penalty is added to the one for voluntary vassals (130t), and they're displayed together as "We are worried about our rivals being vassals to your empire".	If a civ has at least one vassal, the "vassals to your empire" penalty is based on the military power of the master and all its vassals. Other than that, there are rank-based penalties (130c).

Rationale	Not fair to give a penalty based on military power only to civs that control a vassal. A civ that just conquers all cities isn't less threatening. Since I don't want to remove the vassal penalty, I'm adding a counterpart that penalizes conquered cities. In any case, penalizing military power isn't good because that hardly affects human civs. The personality-based term means that e.g. Genghis Khan won't easily get upset about expansionism, which is consistent with his own behavior and his first-impressions bonus with other warmongers. Moreover, I don't want all leaders to increase the penalty synchronously because, then, conquering a single city could have a big negative impact on relations overall; the personality factor makes sure that the overall effect changes smoothly. Power ratio is factored in because it doesn't help if small civs stop trading with the expansionists – this mostly hurts the small civs. As razing doesn't remove tile culture, this penalty can't be circumvented by razing (and razing carries its own diplo penalties too). I've considered ignoring cities culturally owned by an enemy (war enemy or worst enemy) of the AI civ that opposes expansionism, but concluded that this shouldn't make a (big) difference; territorial expansion needs to be opposed even if it has happened at the expense of an enemy.
See also	Without 099 (culture of dead civs stays in the game), this change wouldn't really work because cities conquered from eliminated civs wouldn't count. CFC post explaining why I've increased the expansionism penalty (though not its upper limit) a bit in AdvCiv 1.01.
Tbd.	Should perhaps be reduced if the AI civ that opposes expansionism has conquered much more territory. Can seem a bit hypocritical the way it is now.

130x	Ideological relations modifiers (religion, civic) take longer to attain the higher they go

See also	130n dynamically adjusts these modifiers to the popularity of the ideology in question, and slows down different-religion penalties.
	148 lowers the limits for the positive ideological modifiers for many AI leaders.

AdvCiv	BtS
The time-based same-religion and favorite-civic modifiers increase by one after 10 turns, by one more after 11 turns, then by one more after 12 turns etc.	They increase by one every 10 turns.

Rationale	Since the modifiers have tighter limits now, I feel that they should also grow a bit more slowlys, so that they don't become too quick to max out. Making them grow more slowly initially could frustrate players, so lets try a non-linear progression.

130y	Lower diplo penalties for wars involving vassals, DP or seeing little action

AdvCiv	BtS
When making peace, the declared-war relations penalty is reduced (but not beyond -2) if the other side has had very little war success, namely less than 0.3 times the equivalent of capturing a city times the era number (0 for Ancient; i.e. full penalty for early attacks on Workers).	The declared-war penalty is solely based on the number of declarations of war; only the (unimportant) hidden "this war is going badly" penalty is based on war success.
The penalty is also reduced if either side is a capitulated vassal. And a capitulated vassal reduces its declared-war penalties when it regains its independence.	When declaring war on a master civ, war against the vassal civs is implied, and the aggressor gets the full (never decaying) -3 "You declared war on us" from each vassal, even if the war takes place entirely in the master's territory, and ends up freeing a capitulated vassal. Likewise, when a master declares war, its vassals suffer a -3 relations penalty.
If a vassal is freed because its master has capitulated to a third civ, the third civ automatically makes peace with the vassal, but without signing a peace treaty.	When a civ capitulates, the vassals of that civ are freed. They remain at war with the master of their former master, and the former master declares war on the former vassal.
If the third civ doesn't have much war success against the freed vassal, and the free vassal had been a capitulated vassal, the third civs gains +2 "You've granted us independence" from the freed vassal.	

Rationale	Shouldn't hold civs fully accountable for declarations of war if these declarations were enforced by the vassal system, especially not for capitulated vassals. The change should make it easier to free vassals and have a cooperative relationship with them henceforth.
	I'm factoring in war success because vassals should not be happy about being liberated when that involves killing most of their units or taking away their cities.
	The reduced penalty for non-vassals practically only applies to wars where neither side sent any units or an invader changed its mind in the face of tough defenses. Not plausible for such wars to have long running repercussions.

See also	130h disables war-on-friend penalties for attacking master/vassal alliances; sha reduces "war spoils our relations" penalty if there is little war success.

Tbd.	Should perhaps also add "granted us independence" memory when the master is eliminated.

A DoW caused by honoring a DP leads to only a -2 relations penalty from the civ that triggered the DP. (No change to "war on friend" penalties.)	Full relations penalty (-3).
Rationale	Diplo penalties for fulfilling a DP seem justifiable, or at least not wrong enough to change radically. I'm just reducing the penalty a bit.
Tbd.	Should probably not assign war-on-friend penalties when war is triggered by a defensive pact, and perhaps not for wars declared by capitulated vassals either.

130z	AI gives help also to other AI civs	
AdvCiv		*BtS*
An AI leader may decide to gift a tech to any non-vassal civ that has fallen behind and that the AI leader is at least Pleased with. Friendly relations, a low difficulty setting and a high Diplo victory stage increase the probability of such a gift. The choice of the tech is biased toward low research cost. (No change to the AI routine for gifting tech to vassals.)		Only human civs can benefit from this. Pleased attitude is required, Friendly and difficulty don't matter. (Although a low difficulty makes the AI easier to please.) The tech chosen uniformly at random.
See also	112 deals with tech gifts between vassal and master. The gifted tech is also chosen randomly based on cost in that context. The part that takes into account the difficulty level is based on 250a.	
Rationale	Especially with SPaH, it's not so unlikely for experienced players to trigger this AI behavior, and I don't think those players would want the AI to help them compete. So one could either disable the help routine for medium and high difficulty settings, or extend it so that AI civs can benefit too. I've done the latter because it can actually make sense for an AI civ to help a weaker partner; humans do that too sometimes.	
Tbd.	Should perhaps ask for gold in exchange if the recipient happens to have some.	

131	Misc. changes to AI evaluation of units, buildings, techs, civics, religions
See also	cdtw and rom for such changes adopted from other mods. 121: AI changes to Worker builds and citizen assignment 042: Anticipate border expansion when evaluating Work Boat 192: AI for first border expansion through culture buildings. Some very minor tweaks are only documented through comments in the code.
Credits	A few minor changes adopted from MNAI are tagged with "advc.131" (and credited to MNAI in the source code).
Tbd.	Open K-Mod issue: "*Fix AI's tech evaluation*" (link) And regarding the Caste System `SpecialistValid` ability (comment in `CvPlayerAI.cpp`): "*todo: the current code sucks. Fix it.*" True, but I'd like to remove that ability at some point (it's about the opposite of what Caste System should do), so I'm not going to fix it.
AI more likely to build a high-utility building than an arbitrary XP or gold building.	

AdvCiv	K-Mod
Prior to the Industrial era, the AI is reluctant to build certain National Wonders in its capital, in particular Moai Statues and Globe Theater.	Moai Statues in capitals are fairly common in K-Mod. If Globe Theater is also built in the capital, the AI can't build Oxford University there.
If a city has a Great Person Point (GPP) rate of at least 7, the AI considers building the National Epic. The threshold increases by 2 with each era beyond Classical.	The threshold is 10 (flat), and the AI often doesn't build the National Epic until late in the game.

Rationale	7 GPP could be two specialists and one wonder or three specialists. Could happen in the capital, which is OK with me; an early "GP Farm" in the capital isn't bad.
Tbd.	Oxford in the capital shouldn't always be the best choice; that needs to be addressed by a balance change.
	Should discourage Moai in any city with a lot of good unworked land tiles. `CvCityAI::AI_buildingSeaYieldChangeWeight` would be the place for that change.
	National Epic is probably still not given enough priority.

AdvCiv	K-Mod
AI tech evaluation: Reduced the value assigned to units whose resource requirements aren't met. E.g. to make the AI less inclined to research Horseback Riding when it doesn't have a Horse.	K-Mod: Already reduced; I'm reducing it more.
AI switches state religion probabilistically based on the ratio of the utility of the new religion to the utility of the current one. Increased attraction to AP.	Current state religion is counted as having between +33% and +50% utility to create inertia and avoid revolutions. However, once another religion reaches higher utility despite this bonus, the AI switches immediately, leading to occasional back and forth switches as cities are conquered and lost again.
The algorithm for AI civics changes should now handle negative utility values correctly.	The inertia mechanism has the opposite effect when dealing with negative utility values. Negative values probably don't occur though (nor in AdvCiv I think).

Rationale	The values of religions can fluctuate when two religions spread in parallel or when city owners change repeatedly in a war. Probabilistic switching means that the AI tends to wait for some turns when the one religion isn't far better than the other.
	I had implemented the same thing for civics but removed it again; see my post here in the K-Mod subforum (the "Update" part under No. 1).
Credits	The change for negative utility values is inspired by a similar change in "RoM: A New Dawn" by koshling. SourceForge revision: link
Tbd.	Can currently only switch religion every 15 turns unless Spiritual (see calls to `AI_setReligionTimer`) and civics every 20 turns. Would be better to lower the per-turn probability of switching based on the turns elapsed since the latest switch.
See also	001r fixes a bug that had caused the AI to switch civics too readily.

The AI only starts a revolution if it has enough gold in the treasury to prevent the strike counter from reaching 2 during anarchy.	During anarchy, civs have 0 income and expenses except that gold deals still need to be paid. This means that anarchy leads to a strike when a civ has no gold in its treasury. If anarchy lasts only one turn, the strike has no immediate effect, but multiple turns of anarchy lead to disbanded units. Moreover, the strike turn counter is never reset, so once there has been a strike, even a single turn of anarchy can lead to disbanding.
Rationale	I've thought about suspending gold deals during anarchy, but then what about the civ that is supposed to receive gold? For a human player, it could be confusing if an AI civ pays no gold during anarchy. If gold is received without having been payed, then this could exploited in multiplayer (team) games.
	Perhaps the strike counter should be reset or counted backwards (probabilistically?) when there is no strike; but then a single turn of strike now and then would have no negative consequences at all.
See also	132b: Vassals with a human master reluctant to switch civics. 133: Gold-per-turn deals are canceled when broke

AdvCiv		*Vanilla Civ 4*
Extended the AI evaluation of the Pyramids ability. In particular, the current Government civic is taken into account.		Just counts a fraction of the AI civics value for each unlocked civic. A tiny fraction, considering that building and civics values aren't on the same scale. K-Mod 1.46 note: "*compare to current civics!*"
Rationale	Was greatly underrated and the AI was building the Pyramids very late (late Classical era usually). Actually had to make the AI underrate the Pyramids again by a factor of more than two after some tests. In BtS, they're something that human civs without much land and mediocre terrain can reliably fall back on so long as they have decent food sources; not good for balance to let the AI build them aggressively.	
See also	This CFC post noting that good wonders are not being prioritized by the AI might refer to the Pyramids. I suppose the BtS AI builds them a bit sooner than K-Mod through non-intelligent factors (flavor, building focus, culture).	

Reduced the tech value counted for buildings when an AI civ hasn't founded a second city yet and the population of the capital hasn't grown beyond size 3.

Don't count tech value for potential domestic trade routes when having only 1 city.

The AI assumes that it is not geographically isolated when more than 25% of its continent remains unexplored. Consequently, naval exploration is not prioritized in the very early game on account of being isolated.

So long as the AI isn't sure whether it is isolated (no one met and 30 turns not yet passed), the values counted for military units are reduced.

Reduced the utility counted for working water when the average city population is small.

Rationale	Mostly to prevent the AI from overestimating Fishing and Sailing in the very early game. Apart from Lighthouse, an early discovery of Sailing can lead the AI to waste time on a Galley or to construct Moai Statues in the capital.

Swordsman and Legionary have Attack City as their default AI type. Non-default AI types are Attack and Reserve.	For all Swordsman units, the default is Attack, non-default Attack City and Reserve. The earliest non-siege unit with default AI type Attack City is Maceman, the next one Grenadier. The default AI type seems to matter mostly when the AI evaluates units that it can't train yet. Will then, for each AI type of the new unit, compare the new unit with the best current unit that has the matching default AI type.
Rationale	The main advantage of Swordsmen over Axemen and other early units is that Swordsmen are better at attacking (AI) cities held by Archers.
See also	907a renames Praetorian to Legionary.
Battleship has the (non-default) AI type Escort.	Destroyer is the only late-game unit with Escort AI type. I don't think the Escort type is required for accompanying cargo ships, but if more escort ships are needed, the AI will only train ships with the Escort AI type to fill the ranks.
Rationale	Battleships don't come much later than Destroyers and aren't that much more expensive. They can handle enemy Battleships, and a mix of Destroyers and Battleships can stand up to the endgame ships (Missile Cruiser, Stealth Destroyer) and air attacks.
See also	905a removes the Escort AI type from Caravel.
The evaluation of an extra trade route granted by a technology takes into account future cities.	Looks like the K-Mod formula only accounts for current cities.
Rationale	The AI is sometimes very slow to discover Currency.
Tbd.	Currency still seems to be underrated (and Code of laws overrated).
AI building evaluation takes into account Bureaucracy when evaluating special commerce modifiers. And some value is added to account for future increases of the city's base commerce.	Special commerce modifiers are evaluated by taking the modifier times the current base rate of the respective special commerce type. K-Mod increases the result to account for possible future weight increases (slider position), and accounts for future growth when evaluating base yield modifiers – but not for special commerce modifiers.
Rationale	The AI had tended to omit the Education tech entirely.
AdvCiv	*BtS*

Ancient technologies and Horseback Riding have an asset value of 6, Classical techs 12, Medieval 18, Renaissance 24, Industrial 30, Modern and Future 36.	8, 16, 24, ...	
The Infantry unit has an asset value of 4.	3 for Infantry but e.g. 4 for SAM Infantry.	
	The total asset value of a civ is computed as 2 per citizen, 1 per land tile, up to 48 per technology, 1 to 5(?) per non-wonder building, 16(?) per great wonder, 8(?) per national wonder. For units, the asset value usually corresponds to the square root of its power, rounded down; though e.g. not for Infantry.	
See also	130p uses asset totals for computing trade-based relations modifiers.	The total is used only for deciding whether an AI civ asks for a free technology or offers one.
Rationale	BtS asset values are heavily skewed toward technologies. E.g. early technologies cost only 60 research and count as 8 assets, whereas 4 population (also worth 8 assets) cost at least 88 food and 8 Archers cost 280 production. This is probably intentional: when the human player gets ahead technologically, the AI is supposed to ask for help even if it has more units and citizens. It was just a bit too extreme, at least for the purposes of 130p.	
	Horseback Riding being treated as an Ancient tech might be deliberate (*shrug*).	
Tbd.	The asset values for buildings look crazy; e.g. 1 for Drydock but 2 for Barracks and 3 for Lighthouse. Hard to say how the original developers came up with these.	
When evaluating the Great Lighthouse, check planned city sites to estimate how many coastal cities there might be in the medium term.	BtS only considers the currently owned coastal cities, K-Mod adds a projection based only on the map size.	
Rationale	Just a little tweak.	
131b	AI weights in `Civ4TechInfos.xml`	
AdvCiv	*BtS*	
Negative AI weight for Feudalism. And slightly increased the cost of Iron Working.	All weights are 0 in BtS and K-Mod. AI weights are added to the utility values computed in AI tech evaluation. The intention was probably to give non-DLL modders a way to bring the AI to research techs with novel abilities.	
Rationale	In test games before releasing v0.93, half of the AI civs were researching Iron Working right after Bronze Working, and Feudalism was practically always researched before Machinery. This is out of step with the historical development and, more importantly, makes these techs unattractive to research for human players as they're easy to get in trade.	
	In testing with v1.0, Iron Working isn't as popular anymore; not sure why. So I've set the AI weight for Iron Working back to 0. Usually some civs go for Iron Working very early – I suppose civs that lack both Copper and Horse –, while others delay it until the late Classical era (if they can't trade for it). I guess that's OK.	

Tbd.	This is a temporary measure. Not sure how to fix this more permanently though. Longbowman and Serfdom really are very useful for the AI.
	The AI weight should arguably be multiplied by the number of cities in the DLL in order to match the scale of the tech utility values. Currently, a non-zero AI weight for a late tech would have less impact on a map that gives civs a lot of room to expand. Shouldn't be much of a problem for early/ midgame techs because civs can't expand that quickly.

131c	Power values of buildings	
See also	UWAI uses its own power ratings (cf. 104e, also for some other changes to power values), but the power graph (Info screen) still shows the BtS power ratings and they're still used for many AI decisions.	
AdvCiv		*BtS*
Only buildings with defense, XP and war weariness abilities contribute (a little) to the military power value (power graph) of a civ.		Some buildings with production abilities also have power values, e.g. Forge.
Rationale	Counting some military power for buildings seems fair enough in general, may help a little to deter the AI (and humans) from attacking difficult targets. Production abilities are reflected by the production curve though and both UWAI and K-Mod's AI_startWarVal function take the production curve into account. (Arguably, they should also take into account XP abilities, but, so far, they don't.)	
Config	`iPower` tag in `Civ4UnitInfos.xml`. Note that the DLL does *not* scale those values according to the game progress; therefore, a late-game building like the Pentagon, contemporary with e.g. power-22 Infantry units, needs to have a high power value to have any noticeable impact. It would be better to let the DLL compute a power value from the various militarily relevant building abilities and adjust that value to the game progress, but this isn't worth the implementation effort (in particular, updating the caches at CvPlayer and CvArea would take some effort).	
See also	This CFC post by made me aware that the Great Wall's power value is too high – especially since 310 usually disables the Great General ability.	

132	Changes to civics and religion trades	
AdvCiv		*BtS*
Can (a) bring vassals and war enemies (peace negotiation), or (b) any civ through a Spy, to switch to any economy and religion civic except the initial ones, and to any major religion (same threshold as for "not enough of our people follow that faith"). Exception: A master with a state religion can't ask its a vassal to accept a different state religion. A civ with a state religion can still (as in BtS) try to force another civ into that religion if the other civ has at least one city with that religion.		Can only ask anyone to switch to own civics (except the initial ones) and own religion (unless "not enough people ...").
The petitioned side charges twice as much trade value in case (a) if the petitioner is not running the target civic or religion. In case (b), twice as many espionage points are needed.		

AI civs refuse trades that ask them to make multiple civics changes in the same column. In multiplayer (not tested), I imagine such trades are possible and will result in the player adopting the civic that was added to the trade offer first.		As the trade table is not part of the SDK, mods can, to my knowledge, block only individual trade items, not combinations of trade items.
Rationale	"Preach only what you practice" is sensible – but only when negotiating as peers. E.g. it makes sense to demand a switch to Pacifism as part of a peace treaty, or to ask a vassal to switch out of Mercantilism or Theocracy so that the master's corporations and religions can spread. (UN votes are yet another matter. I think there are few, if any, civics beyond the current UN civics that would work well and make sense.)	
	Also more interesting (and entertaining) options this way.	
	A higher cost for switching to an unused civic makes sense in case (a) because the other side will suspect bad intentions, and (somewhat) in case (b) because the Spy owner lacks familiarity with the target civic/ religion.	
	Why not allow all civics? Don't want that many options on the diplo screen. Also seems far-fetched to switch someone to, say, Vassalage in the Modern era, when no civ in the game has been in Vassalage for 500 years. Assume civs to be somewhat flexible in their religion civics and economics. These are also the columns with the isolationist civics Mercantilism, State Property and Theocracy.	
Config	New tag `bCanAlwaysForce` in `Civ4CivicInfos.xml`.	
The force-religion espionage mission requires the target religion to be present in the city where the mission is executed.		The target religion has to be present in any one city.
Rationale	More plausible this way; don't think it matters much for game balance or AI behavior.	
Increased the impact of the number of cities with the target religion on the mission cost.		K-Mod already adjusts the mission cost but rarely increases it by more than 100%.
Increased the base cost of force-religion and force-civic by 25%.		
The AI doesn't use the force-religion mission if the target religion has few adherents (same threshold as the "too few of our people follow that faith" response).		The AI uses the mission even if just 1 city has the target religion. The affected civ usually switches back to its former religion after 5 turns.
Rationale	Being forced to convert from a widespread religion into one with just a single city can be very painful. I've considered allowing only major religions as the target religion, but a cost increase and a change to AI behavior should suffice.	
Config	Partly in `Civ4EspionageMissionInfos.xml`	
Tbd.	The number of religious cities should be factored into the mission cost *modifier*, not the base cost, because the UI shows a breakdown of the modifier, i.e. the impact of the city count could then be displayed by `CvGameTextMgr::setEspionageCostHelp`. Same for the cost increase when the Spy owner doesn't run the target civic or religion.	
When asked to change civics or religion, the AI predicts the losses from anarchy based on the goods-produced (production) and GNP (gold + research; 004s) curves.		Based only on city counts, both of the AI civ and the human who's asking for the change. Can underestimate the cost by a factor of 10 in the late game.
Increase the multiplier for the difference between the current and the new AI civic value.		The cost of running a bad civic is also underestimated by a factor of 10.
Rationale	The human cities shouldn't play a role, and the AI city count could well stay the same	

	during the second half of the game, but the cost of anarchy doesn't.	
Tbd.	The AI should charge extra or refuse when asked to switch out of a hurry-production civic by a player who may well be about to declare war. Or perhaps the change-civics attitude threshold already covers that wariness … Different tack: Tack a peace treaty onto change-civics and change-religion trades.	
132b		
AI vassals are reluctant to switch civics if their master is human.	Vassal/master status doesn't affect civic choices. The diplo bonus from sharing a religion with the master factors into the religion choice of the vassal.	
Rationale	Humans tend to ask vassals to adopt certain civics; switching back and forth doesn't help either side.	
132c	Switching to a state religion requires at least one city with that religion; doesn't suffice if another team member has a city.	
Rationale	No special need for team members to align their state religions. Major religions are also determined based on per-civ city counts, and not per-team.	

133	AI cancels more deals	
AdvCiv		*BBAI*
AI cancels tribute deals ("it's time for your tribute") once a vassal agreement ends.	Tribute deals remain in place until (if ever) the AI reaches so much military power that it wouldn't have agreed to the deals in the first place.	
Rationale	The problem of lingering tribute may have been introduced by BBAI, which replaced BtS/Warlords custom code for deal cancelation with calls to `AI_considerOffer`. K-Mod has mostly repaired this (deals from non-vassal tribute demands and gifts do get canceled), but missed a spot.	
When deciding whether to cancel a resource trade, all trade-denial conditions are checked.	Trade value is checked every turn, but not denial conditions like those based on attitude. There's a special clause that cancels deals with the worst enemy of an AI civ.	
Rationale	I think this was the intended behavior from the beginning, but a bit awkward to implement. (Comment by karadoc in `CvPlayerAI.cpp`: "*getTradeDenial is not equipped to consider deal cancelation properly.*") The denial check also cancels deals that no longer make sense for the other side, e.g. when a player imports Rice and later settles on a Rice resource.	
AdvCiv		*K-Mod*

The tolerance for continuing resource deals when the trade values have become uneven starts at 50% and decreases to 20% over time. Even if the tolerance is exceeded, deals with human players aren't necessarily canceled. A cancellation probability is computed based on the ratio of the trade values, the absolute difference of the trade values in relation to the AI civ's total per-turn commerce, the game speed setting and the number of civs that the human player is in contact with.	10% in BtS, 25% in K-Mod; doesn't change over time.

Rationale	The changes to AI resource evaluation (change 036) lead to more fluctuation in trade values; hence the increased tolerance and randomness
Credits	xyx and Bestban made me aware of problems with the formulas I had used in AdvCiv 0.96e and earlier. CFC link
See also	Relies on code written for 036. 155 allows the AI to cancel deals with members of its own team.

AdvCiv	*BtS*
When an AI civ cancels several resource deals with a human civ at once, only one renegotiation popup is shown. That popup lists the trade items of all canceled deals.	A separate popup for each canceled deal.

Rationale	Mainly to make sure that the human player is aware of all canceled deals when renegotiating.
See also	Of some importance for 074 (show must-be-joking resources on the trade table if they've just been canceled).

If an AI-AI resource deal is canceled, the AI immediately tries to make a new deal.	Depending on the AI personalities, regardless of cancelation, a resource deal between two AI civs is considered every 1-5 turns on average.

Rationale	Interrupted resource deals can lead to some back and forth in the city management of the AI. Not sure how harmful that is, but better to avoid it. Should also lead to more stable trade relations that are easier to keep track of for human players.

When a vassal agreement ends, all remaining deals between vassal and master can be canceled by either side (turns-to-cancel set to 0).	Deals with a minimum duration can't be canceled until the 10 turns are over, regardless of the vassal agreement. This includes resource gifts to the vassal.

Rationale	A minor issue. They shouldn't be forced to continue free resource deliveries. For players, it can be confusing to see free resource trades continue between two AI civs that no longer have a vassal agreement.

AI cancels Open Borders, Defensive Pact and Permanent Alliance when AI attitude drops below the thresholds for signing the respective agreements. Cancellation is delayed with a per-turn probability except when the other side has become the AI civ's worst enemy.	AI cancels these agreements immediately when the other side becomes its worst enemy. Otherwise, AI attitude doesn't lead to cancelation (though other considerations may). Borders can remain open indefinitely despite Annoyed attitude.

Rationale	I think the original idea was that deals get canceled when their attitude threshold isn't met anymore; they just didn't get the implementation right. The delay is supposed to give the other side time to amend relations before canceling OB. (Once OB are canceled, it's difficult to get them back because the "brought us together" diplo bonus is lost.)

The AI cancels gold-per-turn (GPT) deals when its total payments to a player exceed the limits that apply during deal negotiation.	The per-player GPT limits apply only when negotiating deals, i.e. the AI will never cancel a deal on this account.
See also	[CFC post](#) describing how the AI GPT limits can be circumvented by "subsidizing" an AI civ with a GPT gift for 10 turns. Note that the same trick isn't applicable for gold paid as a lump sum – because the AI doesn't subtract cash received from cash paid (whereas the AI does consider the balance of mutual GPT payments). [036](#) lets the AI evaluate the usefulness of resources – arguably making the GPT limits dispensable –, and relaxes the GPT limits. Still: I don't want the relaxed limits to be circumvented.
(Not an AI change:) When a civ doesn't have enough gold to pay for an active gold-per-turn deal, not even after (force-)increasing the gold slider to 100%, the gold-per-turn deal gets canceled by the game rules.	A civ that can no longer pay for gold-per-turn deals (typically after losing cities at war) will go into strike. The recipient keeps receiving the gold even if it isn't paid. The AI will cancel gold-per-turn deals, but only once they're no longer too recent to cancel.
Rationale	It's not uncommon for small civs in the late game to pay a large portion of their budget for imported resources. When a big (human) civ quickly takes cities from such a small civ, the effect of the subsequent strike can be noticeable and confusing. It's also illogical that the gold keeps getting received without getting paid. I don't see a credible way how this new rule could be abused (i.e. to cancel war reparations).
See also	This situation might be exceedingly rare if it weren't for the changes to AI resource trades ([036](#)).

134	Changes to AI-to-human offers
See also	[136b](#) also fits here (map trades offered by the AI) [550](#) deals with tech trades.
134a	Workaround for a bug in the EXE that prevents AI peace offers
See also	[001e](#) deals with similar bugs.

AdvCiv	*BtS*
The AI offers peace and capitulation to human civs based on the `ContactRand` and `ContactDelay` values in `Civ4LeaderHead Infos.xml`. I've changed the values of some leaders: `Delay` decreased to 8: Gandhi `Delay` increased to 15: for eleven leaders `Delay` increased to 20: Sitting Bull, Tokugawa `Rand` decreased to 10: Gandhi `Rand` decreased to 12: Mansa Musa, Pacal, Joao, Hatshepsut, Ashoka, Huayna Capac `Rand` decreased to 15 for sixteen leaders	The AI never contacts a human player to offer peace or capitulation. I think in Vanilla Civ 4, peace offers still worked. Perhaps the Warlords developers introduced the bug when adding capitulation offers (and broke the capitulation offers too), or some patch is responsible, but the BtS expansion seems like the likeliest culprit. The contact values were 20 (`ContactRand`; i.e. 1 chance in 20) and 10 (`ContactDelay`) for all leaders.

`Rand` increased to 20 for seven leaders The contact timer is reset when peace is made. The [UWAI](#) probability of peace offers also takes into account war utility. Note that, if UWAI is disabled, the `ContactRand` values also affect the timing of peace offers between AI civs. I've added some checks at the start of the human turn to verify that the peace offer still makes sense for both sides; if it doesn't the offer is silently discarded and the AI contact timer is reset, meaning that the AI can make a new peace offer directly on the next turn. To work around the error in the EXE, I'm having the DLL feign peace between the two civs at just the right moment. This has no observable side-effects (assuming that I've implemented it correctly).	The contact timer isn't reset, but the delay value equals the peace treaty duration, so having contacted the human player in one war can't really affect contact behavior in a subsequent war. It seems that, before displaying any AI trade offer, the EXE (class `CvDiplomacyScreen`?) verifies that the two civs are either not at war or that the trade includes an item that implies peace. The latter part is somehow not correctly implemented.

Rationale	I guess I fixed it mainly because it was challenging to do. AI trade offers (including peace) don't have an important function in the game, though they're nice to have for flavor. As for AI personalities, I'm setting lower `Rand` values for leaders whose historical counterparts have actually sued or peace. Higher `Rand` values for especially dogged personalities. Higher `Delay` for some leaders to ensure that they don't appear to be groveling by asking for peace repeatedly.
See also	K-Mod has introduced a similar discount for other AI offers. A good idea; without such an incentive, the AI might as well not contact human civs at all. See the 2nd page of the K-Mod [chapter](#). CFC [thread](#) about the lack of AI peace offers in BtS. Humans can't currently receive multiple peace offers in one turn. Moving AI diplomacy to the start of the human turns (see Tbd. under [001e](#)) would fix this.
Tbd.	In networked multiplayer, the peace offer sometimes seems to get through without checking at-war status. I haven't seen this lead to any problem, but I'm not sure what's going on there. UWAI sends a capitulation offer as soon as the AI becomes willing to capitulate. I don't think this totally obsoletes the capitulation Civ4lert, but it would be nice if the alert could be suppressed when a capitulation popup has just been shown.

AI peace offers can come with a discount. However, if a peace offer is rejected, AI war utility is slightly increased (by up to 4 depending on the AI leader's contact delay value) until the contact timer is back at 0, i.e. for 8 to 20 turns.	No discount, no adverse consequences for rejection.
The AI adds to capitulation offers whatever trade items it is willing to give based on war utility.	When offering capitulation, no reparations are offered.

See also	K-Mod has introduced a similar discount for other AI offers. A good idea; without such an incentive, the AI might as well not contact human civs at all. See the 2nd page of the K-Mod [chapter](#).

Tbd.	Could add an AI memory type with a -1 relations modifier to communicate the effect of a rejected peace offer more clearly. But I can't come up with a snappy phrase for the explanation text; "You rejected our peace offer." is too lame.

134b	(Disabled) No discount if recently begged	
AdvCiv		*K-Mod*
~~The AI offers no discounts to civs that have recently asked for a gift.~~ I'm afraid it's usually more profitable to ask for gifts all the time than to wait for discounts. If so, this change would mostly disable discounts, which isn't what I want. (I want to disincentivize periodic gift requests.)		The AI doesn't grant gift requests by civs that have recently asked for a gift, but discounts aren't affected by gift requests.
Rationale	An attempt to disincentivize gift requests by the clock. Some players set alerts for this every 25 turns; see e.g. this CFC thread.	

135	Changes to multiplayer	
See also	004v removes square brackets around the names of human civs on the scoreboard. 108b applies `StartingLocPercent` from handicap in multiplayer. 054 removes some options from the Staging Room screen.	
No Civ4lerts about humans willing to become a vassal. AI won't ask one human to stop trading with another human.		
AdvCiv		*BtS*
135a	Resource bubbles	
If a player activates resource bubbles in Hotseat, they remain active until a player deactivates them.		Resource bubbles are deactivated at the end of each human turn. Other map layers (e.g. yield display) are not reset.
Rationale	The BtS behavior is annoying if all players want to play with resource bubbles. Otherwise, the players could still agree not to activate them, or one player flicks them on and the next one flicks them off.	
Tbd.	The proper solution might be to remember each player's active layers at end of turn, and restore them at the beginning of the next turn of that player. Same for player options and BUG options, which are currently shared by all players. Could store layers and options in savegames, but storing them in non-serialized data members of `CvPlayer` would already be an improvement.	
See also	004m increases the default zoom distance (also reset each turn in Hotseat).	
135b	`MoreCiv4lerts` in Hotseat	
AdvCiv		*BUG*
Added per-player memory to the BUG alerts in the `MoreCiv4lerts` package. (The most useful ones, like tech trades, are in that package.)		`MoreCiv4lerts` fire every turn in Hotseat; unusable.
Rationale	The first package of alerts seems to have been implemented with multiplayer in mind but not the "More" ones. This might also be fixed in the BUG main branch (see this commit).	
Tbd.	Don't know if the alerts also work for networked multiplayer.	

See also	106c also fixes issues with Civ4lerts.
	106b always opens the Event Log when there is a new message in Hotseat.

135c	Debug tools in multiplayer

AdvCiv	BtS
Can use WorldBuilder (WB) and Debug mode in multiplayer if `ENABLE_DEBUG_TOOLS_MULTIPLAYER` is set in `GlobalDefines_devel.xml`. (I guess all players should set it if multiple PCs are involved.) In networked multiplayer (i.e. anything except Hotseat), "chipotle" needs to be set **as the game name** under "Host Game" in addition to the XML switch. The text on the large flag button will then say "cheats enabled" in order to make sure that no player can secretly cheat. The cheat code in `CivilizationIV.ini` doesn't affect multiplayer. Entering the WB makes network games go out of sync (OOS) immediately; will have to save and reload afterwards.	WB is always available in singleplayer; Debug mode (Ctrl+Z) requires cheats to be enabled through `CivilizationIV.ini` (code "chipotle"). In multiplayer, both WB and Debug mode are blocked through a cheats-enabled check outside the SDK and several checks inside the SDK, presumably to make certain that cheating is impossible.
If the game name is "chipotle", only players with even slot id numbers create autosaves.	When testing multiplayer on a single machine, both players try to write to the same autosave file, leading to annoying error popups.
If a player changes the game name to "chipotle" through the in-game menu ("Game Details"), an announcement informs all players that debug tools have been enabled.	

Rationale	Debug tools are indispensable for multiplayer testing. Not sure if the OOS issue with the WB can be fixed; could just be a side-effect of my hack to make the WB accessible.
	Still missing: Python console (impossible?)
	Autosaves: Don't know how to check if both instances run on the same machine, so I'm assuming that the game name "chipotle" is only used for such test games.
See also	127 allows enabling AI Auto Play in multiplayer.
	001n fixes OOS errors.
	007 describes how OOS errors can be debugged on a single PC; and there's a short guide about the various log files.

To get rid of the cheat level checks in the `CvGameTextMgr` class more easily, I've split the `setPlotHelp` member function into several subroutines and made some changes to const qualifiers. That could make it a bit difficult to merge this change into another mod.

135d	In multiplayer games, the Settings tab (Victory screen) shows the game end turn only if it's different from the default (500 on Normal speed).
Rationale	Tidiness.

136	Changes to map trades
136a	Circumnavigation checked at end of turn

AdvCiv	K-Mod

The game checks at the end of each (human or AI) player's turn if that player has circum-navigated the globe. Thus, if, through a map trade, two players meet the circumnavigation conditions on the same turn, only that player is rewarded on whose turn the map trade is completed. In a human-AI map trade, the human always wins because trades involving humans are always completed on a human turn. This was also the case in BtS (but not in K-Mod). The extra movement still applies to to a whole team. So long as no one has achieved circumnavigation, the AI refuses to trade its map if it can train Caravels. ("We have our reasons.") If it can't train Caravels but the other side can, the AI charges twice as much for its map.	Circumnavigation is checked during each team's turn, not during the player turns. In BtS, the team turns happened all in a row, i.e. without intervening player turns, at the start of a game turn. In K-Mod, team turns and player turns are interleaved, i.e. circumnavigation gets checked for team X right before the players on team X take their turns. That means, when two players satisfy the circumnavigation condition through a map trade, the player on whose turn the trade is completed misses out (always the human when it's a human-AI trade). That player even misses out if it has completed circumnavigation through unit movement earlier on the same turn (before making the map trade). Whether an AI is willing to trade it's map depends entirely on the leader's attitude threshold.

Rationale	The K-Mod timing leads to the trap sketched on the upper right. Checking at end of turn instead puts the AI at a bit of a disadvantage because human-AI trades can't be completed on an AI turn. That said, the AI suggests map trades to human civs relatively rarely, and never with an intention of completing circumnavigation, so the change doesn't really hurt the AI. The trade refusal should make it harder for a human to steal circumnavigation from an AI that is already close.

136b	AI doesn't pester human with unattractive map trades

AdvCiv	*BtS*
The AI proposes no trades to a human civ where the human side receives only the AI map when the human value of that map is 5 gold or less. Changed the rounding of AI trade values to multiples of 5. Makes the implementation of the above change a bit easier. The AI refuses to accept trades with a value of less than 10 (the equivalent of 5 gold).	The AI occasionally asks human civs to exchange maps even if the AI map has no value to the human. Trade values are rounded to a multiple of 10.

Rationale	These map trades are pointless distractions.
Config	The rounding change is implemented in `GlobalDefines_advc.xml`.

AdvCiv	*K-Mod*
No low-value map trades between AI civs either.	I think K-Mod added calls to `AI_counterPropose` to AI deal negotiation. That function will usually insert the map of one side into the deal.

Rationale	Going through all tiles on the map is slightly costly (though not nearly as costly as `CvPlot::updateRouteSymbol` for a human player's map).

137	Default/ recommended player counts; help text for difficulty levels

AdvCiv	*BtS*

On the Custom Game screen, the dropdown menu for the sea level includes recommended changes to the number of players. ("+x% players recommended")	When the world size is changed, some player slots are automatically opened or closed in order to match the recommended number of players. No such thing happens when the sea level is changed.
Rationale	Having the proper number of civs is imo pretty important, and sea level has a big impact on this. The recommendations, if followed, should lead to about the same number of cities per civ as with Normal sea level. Recommendations weren't my first choice; I had wanted slots to be opened and closed automatically when the sea level changes. This can't be done because the Custom Screen isn't part of the SDK. When slots are opened/closed in response to a world size change, it's possible (through a hack) to also factor in sea level, but then the sea level needs to be configured before the world size, which isn't what players normally do.
Changed the default player number for Standard-size maps to 8 (+1), Large maps to 11 (+2) and for Huge maps to 16 (+5). To fine-tune the space per player, I've also reduced the default grid dimensions for some map sizes a little bit: Duel 36x24, Tiny 48x32, Large 100x64, Huge 124x80. This results in the following ratios (width * height) / n (where n is the default player number; i.e. tiles per player): Duel: 432 Tiny: 512 Small: 546 Standard: 573 Large: 582 Huge: 620	Default player counts are Normal - 7, Large - 9 and Huge - 11. Default dimensions: Duel 40x24, Tiny 52x32, Small 64x40, Standard 84x52, Large 104x64, Huge 128x80. Tiles per player: Duel: 480 Tiny: 555 Small: 512 Standard: 624 Large: 740 Huge: 931
Rationale	It's probably intended that larger maps leave more room for expansion, but the differences shouldn't be anywhere near that great. Settling lots of unclaimed land is a drag regardless of map size and shouldn't be the default game experience. Especially for players new to the mod who want to get an impression of the UWAI changes. (As for which is more challenging, protracted confrontation probably helps the AI on high difficulty settings. That said, a higher number of civs increases the chances of some civ getting especially lucky with its starting position, and usually, the strongest adversary presents the main challenge.) The choice for a low player count on Huge maps may have been made for performance reasons; less of a problem with current hardware. Until AdvCiv 0.98, I had also reduced the grid size on Standard size. This may have been closer to the average city count per player that I had in mind, but players with less than average room for expansion were too often compelled to start an early war. (Early war is good in my book, but there should be a choice, at least for human civs.) Ideally, I'd like each civ to be able to found 5-point-something cities near its starting site; enough to stay competitive into the midgame but not the endgame.
Config	`Civ4WorldInfos.xml` Also reduced the `STARTING_DISTANCE_PERCENT` parameter through `GlobalDefines_advc.xml` to match the increased player density. Can't expect starting sites to be as far apart anymore.

See also	The number of *resources* per player increases more slowly with the default player count than the number of tiles. This is because 129 places fewer resources per tile when there are many eligible tiles. 140 adjusts various map size modifiers to match the new player count recommendations. 165 tweaks the dimensions of some particular map scripts to make them work better for the default player counts. 031 fixes an issue in the formula for the target distance between starting sites.

AdvCiv	*K-Mod*
Rewrote the English and German help text for each difficulty and speed setting and [unused] for each map size and sea level setting.	The help text for the sea level and map size settings doesn't seem to be shown anywhere. Help text for difficulty and game speed settings is shown on the final screen ("Snapshot") of the "Play Now" dialog chain. Some of the info is misleading, e.g. it says that Monarch difficulty makes the AI "much smarter" or that Marathon has an entire game's worth of turns in every era.

Rationale	I imagine that most players of this mod (wisely) never use "Play Now", but players that do use it might actually believe the info provided in the help text. My texts about the difficulty settings are a bit clunky but at least somewhat informative and not highly misleading (I hope).

Quests with requirements that scale with the map size (i.e. pretty much all quests?) are based on the map size's building class prereq. modifier. Except for the Blessed Sea quest, which is based on `iTargetNumCities` as in K-Mod.	Quest requirements scale based on the default player count; 2 (Duel) to 11 (Huge) in BtS. The building class prereq. modifier is used e.g. for the Library requirement of Oxford University. It's between +0% (Duel) and +75% (Huge).

See also	154 rounds to the nearest integer when applying the building class prereq. modifier; this is also done for quest requirements. CFC post (last quote box)
Rationale	Now that the default player counts are increased (by a lot on Huge maps), the BtS scaling gets totally out of whack. It was already too sensitive to map size in BtS, and it's a bad idea in general because putting more players on a map gives each player *fewer* cities and thus makes it harder to fulfill quest requirements. The building class prereq. modifier seems like a natural fit for this purpose.
Config	All quests with a map-size adjustment (except Blessed Sea) now call a function `worldSizeTarget` in `CvRandomEventInterface.py`. The formula can be adjusted there.

138	Religion assignment when starting in later eras
AdvCiv	*BtS*

When starting in an era other than Ancient, the automatically founded religions are assigned based on handicap, Spiritual trait and favorite religion. More specifically, on difficulty settings above Noble, the game prefers to assign religions to AI civs, and on lower difficulty to human civs; always prefers Spiritual leaders and AI leaders whose favorite religion is among those to be founded.	After assigning the same number of religions to everyone, the remainder is assigned to a random subset of civs. In particular, favorite religions don't play a role (except possibly with the choose-religions option).
The top priority is still to give each civ the same number of religions. The above change only applies when that is impossible.	

Rationale	Religions are hardly shared in games starting in in the Medieval era or later because almost everyone founds a religion. Assigning the religions to civs that like to spread them (Spiritual, favorite religion) should lead to more block-building like in normal games (Ancient start).
	Also want to make it more predictable for human players whether they'll get a religion if there aren't enough to go around: On Emperor difficulty or higher, probably not. Otherwise, choosing a Spiritual leader should guarantee a religion.

139	AI evacuation of cities	
AdvCiv		*BtS*
When a city is very likely to be lost before the next turn, the AI may evacuate some of its combat units from the city. Units that don't receive defensive modifiers usually leave, and dedicated defensive units usually stay. For other units, the tile defense and remaining city defense can make a difference. Won't try to evacuate if the path to safety looks too dangerous. The AI is less willing to abandon major cities than unimportant ones.		The AI only evacuates noncombatants; may even reinforce hopeless cities.
Rationale	Evacuating units without defensive bonuses and badly injured units seems like a clear improvement. Units with high defensive bonuses can be expected to cause losses to the attacker even when badly outnumbered; might, in particular, draw out the attacker's siege units.	
	It's important that a Stack of Doom can't scare away all defenders from several cities, and settle for peace when the evacuated defenders gather for a last stand. Sounds fairly realistic, but, given the other dynamics of the game, would make conquests too cheap.	
	Fixes the "meatgrinder" loophole, which was still on the BBAI to-do list when the mod was discontinued.	
Config	`AI_EVACUATION_THRESH` in `GlobalDefines_advc.xml`	
See also	107 (more offensive Area AI) could help reclaim evacuated cities.	
	159 tweaks the AI stack strength evaluation, which is the basis for evacuation decisions.	
The AI tries not to leave behind more than 4 units (but there is no hard limit).		

Rationale	A small group of well-fortified defenders should be enough to force the attacker to use siege units; larger groups aren't worth sacrificing.
Credits	Reported about the AI sometimes leaving sizable stacks of defenders behind; e.g. by Cruiser76: CFC post

The safety check at the start of the AI turn sequence anticipates promotions and units in production that will appear at the end of the turn. When deciding whether to let a Great Person join a city or construct/ hurry a building there, the AI skips all cities that are seriously threatened by hostile units.	There is some code presumably with the same intention, but it only checks for hostile units inside the potential target city – where none can exist.

Credits	Prompted by this CFC post by Elkad.

After each attack by an AI unit, the unit owner checks if the attack has provided relief to any nearby threatened city – by repeating the safety check for that city. (This will matter very rarely.)	

Reduced AI trade value for evacuating cities.	The tactical situation does not factor into the AI trade value of a city.

Rationale	The AI should be more willing to give a city away when it thinks that it can't hold onto it, and the recipient (which may or may not be the civ about to conquer the city) should be less willing to pay for such a city.
See also	ctr deals with other changes to city trade value.

The AI stations ships only in cities that, for the moment, appear safe from conquest. Also prefers to use such safe cities for upgrading ships. If no city looks safe, idle ships are kept at sea.	BtS stations ships in cities that are either threatened by no hostile units at all or by fewer units than the ship's current tile. Some idle ships are used for guarding resources and patrol, but the rest is moved into some city even if none are safe. Upgrade cities are chosen only based on proximity, safety doesn't matter.

See also	In K-Mod, an interaction between a K-Mod and a BBAI change causes idle ships to be stationed in the nearest city regardless of danger. 001s deals with that issue and similar ones.
Rationale	The careless positioning of ships can hurt the AI greatly against savvy players.
Tbd.	Should perhaps use a greater maximal range when checking for potential attackers, is currently 3 tiles. Embarked attackers are counted but could be outside this range (still. able to reach the city). Rather than just flagging cities as safe or evacuating at the start of a turn, a scalar safety rating should be (pre-)computed. While gathering forces for a naval landing, the AI currently loads units into cargo as soon as they reach a tile with a ship; then the units stay in cargo for several turns. This is a needless risk.
See also	This change doesn't cover threats from civs that the AI isn't at war with, i.e. surprise attacks on the AI navy on the same turn that war is declared. 162 addresses these. advc.ctr reduces the trade value of cities based on their safety/ evacuating status.

The AI only spends Great People (super specialist, special building, hurry production, Great Artwork) on cities that, at the moment, appear safe.	No such safety checks, neither in the BtS code for Great generals nor in the K-Mod code for the other Great Person types.
Credits	Elkad made me aware of the problem in [this](#) CFC post.

140	Map-size adjustments (`Civ4WorldInfo.xml`)	
See also	[910](#) handles the research modifier based on map size.	
	When Civilopedia is accessed from the main menu, [004y](#) shows numbers that depend on the map size as a range from Duel-size to Huge.	
AdvCiv		*BtS*
Replaced all calls to `CvMap::maxPlotDistance` with a new function `CvMap::typicalDistance` that is less sensitive to world-wrap options (flat/ cylindrical/ toroidal) than `maxPlotDistance`. Maintenance, espionage mission costs and religion spread are affected by this change. My formula is rather complicated; apart from map dimensions and world-wrap, the crowdedness of the map in terms of the (initial) player count is taken into account: The more land is available per player, the lower the distance maintenance per city. Sample numbers from the new formula (Fractal): Small, cylindrical, low sea, 7 civs: 43 Small, flat, low sea, 7 civs: 48 Standard, cyl., 7 civs: 60 Large, cyl., 10 civs: 71 Large, cyl., 6 civs: 92		`maxPlotDistance` is the maximal possible distance between two tiles. That's e.g. 55 on Small cylindrical, 72 on Standard cyl., 89 on Large cyl. and 82 on Small flat. I.e. the world-wrap type has a big impact on distance maintenance.
The distance maintenance multiplier set in `Civ4WorldInfo.xml` starts at 60% on Duel maps and increases in steps of about 7.5 percentage points to 97% on Huge maps.		50% on Duel, increases in steps of 10 to 100% on Huge.
Rationale	Map diameter shouldn't be so decisive. Empires tend to be circular even on oblong maps or maps without world-wrap. What leads to high distance maintenance is mostly a high number of cities, and if a map has lots of space for each civ, civs shouldn't be punished for filling that space.	
	Flat maps are [known](#) as a trick for reducing distance maintenance; on high difficulty settings, AI civs pay much less for city maintenance than human civs, and thus the AI civs benefit less from additional maintenance reduction.	
	I'm not sure how to set the distance maintenance multiplier in XML; that multiplier and the one based on map diameter cancel each other out to an extent. Usually, BtS adjusts too strongly to map size, so I've reduced the step from one map size to the next a bit.	
Tbd.	Not sure if `DistanceMaintenancePercent` in `Civ4WorldInfo.xml` should be changed as well. Currently 50% for Duel and increases in steps of 5 to 100% for Huge. This is countered by the `maxPlotDistance` formula, but only partly I think.	

See also	The Tides of War mod also changes the `maxPlotDistance` formula; [this](#) Kek-Mod Git commit merges that formula into K-Mod. I haven't looked at it in detail, but I think I'm good with my own formula.
The maintenance distance (from nearest government center) is now capped at around 25 on Normal settings; lower cap on smaller maps (`lower maxPlotDistance`), higher cap on larger maps (`higher maxPlotDistance`).	There is an XML parameter `MAX_DISTANCE_CITY_MAINTENANCE`, but this only serves as a weight on maintenance costs; there is no actual limit on the distance or the cost.
Rationale	If a city is very far away from a government center, it shouldn't matter how far away exactly. The BtS cost punishes (very) remote colonies too much.
Config	The distance cap is tied to `MAX_DISTANCE_CITY_MAINTENANCE` (in `GlobalDefines.xml`); changing that value will also change the cap.
The map-size multiplier for number-of-cities maintenance is 42% on Duel and decreases in steps of 4 to 22% on Huge. It gets multiplied by a (hardcoded) factor that slightly increases maintenance on crowded maps (more players initially than the Custom Game screen recommends based on map size and sea level) and slightly decreases maintenance on sparse maps.	45% on Duel, 20% on Huge, steps of 5. Sea level and initial player count don't matter,
See also	Prior to AdvCiv 0.97, I had decreased the modifier in steps of 3. [CFC discussion](#) about that. Also addresses the crowdedness factor. [advc.exp.1](#) deals with number-of-cities maintenance for very large empires.
Rationale	Steps of 5 percentage points would be OK if Duel was at 100%; as it is, the ratio e.g. between Small (35%) and Large (25%) is too large. Or perhaps it would also work if only 9 civs were placed on a Large map, however:
See also	[137](#) changes the default number of civs per map size.
Can draft 2 units per turn on Duel-size and Tiny maps, 3 on Small, Standard and Large, and 4 on Huge.	1 on Duel, 2 on Tiny, 3 on Small and Standard, 4 on Large, 5 on Huge.
Reduced the number of free bonuses from national wonders (Broadway, Rock'n'Roll, Hollywood) by 1 on Standard, Large and Huge maps.	5 on Standard, 6 on Large, 7 on Huge.
War weariness multiplier 137% on Duel, decreasing in steps of 12.5 to 75% on Huge.	150% on Duel, decreasing in steps of 20 to 50% on Huge.
Trade profit multiplier 70% on Duel, decreasing in steps of 7.5 to 33% on Huge.	80% on Duel, decreasing in steps of 10 to 30% on Huge.
Rationale	As above; don't expect the average number of cities per civ to be that much higher on larger maps than on smaller ones.
Tbd.	Corporation maintenance decreases rapidly with increasing map size: 400% on Duel, 50% on Huge. That's probably not ideal.
The six national wonders with prerequisite buildings require 4 buildings on Duel-size and Tiny maps, 5 on Small (times 10/8 percent), 6 on Standard (11/8 rounded up), 6 on Large (12/8)	The map-size multipliers are 4/4 on Duel and Tiny, 5/4 on Small (no change), 6/4 on Standard, 7/4 on Large and 8/4 on Huge, always rounded down.

and 7 on Huge (14/8 rounded up). Cathedrals and the Statue of Zeus require 3 Temples/ Monuments on Small maps.	Consequently, Cathedral and Statue of Zeus, which have a base requirement of 2 buildings, require only 2 buildings on Small maps.
Rationale	Oxford University is almost indispensable for competing in the Industrial era, and e.g. on a Large map with 11 civs (now the default number), it's difficult to acquire peacefully 7 cities where a University makes sense. The change to Cathedrals and Statue of Zeus is just a side-effect.
See also	310 gives the Great Wall a (base) requirement of 1 Wall.
Config	The base numbers of prereq. buildings are set in `Civ4BuildingInfos.xml`, but I didn't change these. The map-size modifiers are in `Civ4WorldInfo.xml`. The change to the rounding behavior is in the DLL (`CvPlayer.cpp`).
Tbd.	Would prefer abilities that scale with the number of buildings, e.g. "Oxford University: +10-15% research (based on map size) per University up to a maximum of 60%". No required number of buildings necessary then.

141	No diplo bonus from gifting GP	
AdvCiv		*K-Mod*
Gifting a Great Person (GP) to an AI civ does not affect relations with that AI civ. Like in BtS, only units with positive production cost can boost relations when gifted.		+1 relations for each gifted GP.
Rationale	Too easy to take advantage of for a diplo victory.	
Tbd.	Gifted military units should only boost relations when the AI has an immediate need for military aid. The relations bonus should probably also be capped; seems exploitable. Comment by karadoc in `CvUnit::gift`: '*It would nice if there was some way this could also reduce "you refused to help us during war time", and stuff like that. But I think that would probably require some additional AI memory.*' I don't think this is worth the effort, but perhaps gifted units should count for "you gave us help" instead of "fair trade". Imp. Knoedel has implemented a new memory type for his Sunset of Civilization mod: Git commit That looks very basic. The main thing of interest would be his `iUnitValue` formula.	

142	Master gets a happiness bonus only from the first peace vassal	
AdvCiv		*BtS*
The master of a vassal civ receives a happiness bonus only from voluntary vassals, and only the first one counts, i.e. at most +1.		Each vassal, capitulated or not, provides +1 happiness in all cities of the master.

Rationale	In large games, the stacking happiness bonus, combined with the free luxury resources from vassals, can let master civs ignore happiness altogether. I also don't like that the bonus applies even if just one city is left; this provides an incentive to keep tiny vassals around, which lead to some oddities in global diplomacy (despite my efforts to fix such issues). Also, oppression of capitulated vassals is hardly a point of pride. Should perhaps remove the bonus entirely, but for voluntary vassals "We influence other civilizations" makes some sense, and these vassals are harder to get. The anger at the vassal side seems sensible (though unimportant) and can't stack. Change 130 and related changes (see under 130v) remove several disadvantages of vassal agreements for the master, so, as far as game balance goes, I don't think a happiness bonus is needed at all.

143	Recently-canceled memory for voluntary vassal agreements (VVA)
If a vassal loses 35% of its territory, it cancels the VVA with a per-turn probability of 15%. Cancels immediately when losing 45%.	Only capitulated vassals revolt when losing territory (50%).
Rationale	The vassal can then make peace independently (possibly by capitulating), while the master is justly punished for not having protected its vassal. The third party (possibly human) is faced with a strategic decision: focus attacks on the vassal or the master? In BtS, only attacks on the master can break up the VVA (by lowering the master's power rating).
Config	`VASSAL_DENY_OWN_LOSSES_FACTOR` in `GlobalDefines_advc.xml`
When a vassal cancels a vassal agreement for no particular reason (i.e. vassal feels safe or powerful enough on its own), the (AI) master remembers this for, on average, 20 turns, and refuses to sign a VVA during that time. When the agreement is canceled because the master hasn't protected the vassal, including the case where a capitulated vassal breaks free after losing too much territory, the vassal remembers this for 30 turns (on average), and refuses another VVA. (Whether the master or the vassal remembers is only relevant when the master is human.)	No memory about canceled vassal agreements. When a capitulated vassal breaks free, it often signs a VVA with the same master on the same turn. When a voluntary vassal is near the power threshold for a VVA, the agreement can flicker on and off every few turns.
Rationale	The flickering isn't only goofy, it also means that war preparations against a former vassal can easily get interrupted by a new VVA, which is frustrating. Also, some mechanism is needed to keep a vassal independent for a while when the master has failed to protected it.
143b	Cancelation and nukes

VVA canceled also after being nuked repeatedly. The threshold is randomized; usually, the vassal doesn't break away until more than half of its cities have been hit. Doesn't cancel if the master has SDI and the vassal does not.	Only lost territory counts (and only for capitulated vassals).
Master's SDI protects vassal.	SDI is a team project, but doesn't protect vassal teams.
Vassal cancels VVA if it has any nukes unless master has SDI and vassal doesn't: "doing fine on our own."	Vassal's nukes factor into its power rating, but a single nuke doesn't usually make a difference.
Capitulated vassals don't build nukes. When a team capitulates, all it's nukes are scrapped. Nukes can't be gifted (between any civs).	Vassals don't hold back on anything. Can gift nukes to circumvent the diplo penalties.

Rationale	SDI is supposed to be an array of space-based lasers. Should be able to intercept any nukes that the SDI owner wants to intercept.
	Vassals that are powerful enough to build nukes should stand on their own.
See also	130v makes the master responsible for nukes fired by a capitulated vassal (obsolete now that vassals can't have nukes).
	112 cancels VVA when near victory and stops capitulated vassals from pursuing victory.
Tbd.	A cancelation condition based on present population vs. population when signed might be better.
	May want to scrap other units upon capitulation too, probabilistically. E.g. when France surrendered to the Third Reich, the French navy was partly scuttled.

144	AI gift request (part of the Dynamic Diplomacy changes)
See also	130v exempts vassals from this change and prevents vassals from granting gifts to rivals or to a disliked master.
	130o deals with memory about tribute demands. Clears recent-demand memory (which is also relevant for gift requests) when war is declared on an AI civ.
	104m: Some gift requests are triggered during war planning.

AdvCiv	*BtS*
Without considering any specifics, the AI refuses gift requests with a leader-specific probability. That probability is based on `ContactRand`: `CONTACT_GIVE_HELP` and between 5% (Gandhi) and 32% (Sitting Bull, Montezuma). I'm capping the probability at 50% for Tokugawa, otherwise he'd always refuse.	The only randomized aspect of the AI response to a gift request is the decay of recent-demand memory: The request is refused if the AI still remembers an earlier request (or tribute demand). So, the first request is handled without randomness, and if a previous request is far enough in the past, it's very likely forgotten (the expected time to forget is 20 turns).
Requests are always refused while preparing war against the player (no change).	`CONTACT_GIVE_HELP` determines only how frequently an AI leader offers a gift to human civs that have fallen behind.

Rationale	Shouldn't be able to use requests to determine reliably if the AI is preparing war. It's semi-decidable now – if a request is granted, then the AI certainly isn't plotting. That said, they might start on the very next turn ... In BtS, `CONTACT_GIVE_HELP` is irrelevant for experienced players; it's nice to give this an actual use.

Memory about requests and rejected demands is not decreased while there is a peace treaty between the two civs. Gift requests are rejected during a peace treaty unless the AI remembers receiving a gift from the player (and that case the peace treaty may well result from a gift to the AI). Likewise, the AI won't ask for a gift during peace treaty unless the human player (i.e. the proxy AI) remembers having received one or when fewer than 4 turns of peace remain. Will only ask for tribute during a peace treaty if it's about to expire (1 turns remaining).	Can decrease on any turn. AI may grant gifts when there is already a peace treaty.

Rationale	To increase the time in between granted requests. A bit too easy in BtS to keep a Pleased AI from planning war: can reliably sign a peace treaty about every 25 turns, which leaves just 15 turns in between, which can sometimes be bridged by asking the AI to attack a third party (which results in a peace treaty; see change [146]). My change makes this tactic unsustainable. Could argue that "begging for peace" shouldn't ever be a tactic. I like that it makes Pleased attitude relevant even for warlike leaders. Think of it as a [Reinsurance Treaty]. I don't like that this is something only humans can use. (Could alternatively merge Kek-Mod's [one-sided peace treaties]. Though communicating those to the player could be a [problem].)

AdvCiv	*K-Mod*
AI refuses gift request if planning war against the player, and refuses with a high probability if war utility is positive (but not yet planning war).	Only refuses if already planning war against the player. (In BtS, the AI doesn't take the resulting peace treaty into account at all; grants request regardless of war plans.)

AdvCiv	*BtS*
When asking for a gift, the AI asks for the tech that it needs most, with a bias for cheap tech.	The requested tech is chosen uniformly at random.

Rationale	Which tech the AI asks for doesn't have to be unpredictable; no need for any random element (though the AI function that I'm using now, `AI_bestTech`, does have a random element). If cost is ignored, the most useful tech tends to be expensive, and I don't want the AI to ask for expensive tech all the time. On the other hand, the AI shouldn't ask for cheap tech often because this makes the decision of the human player too easy. Hence tech cost is given some weight but not full weight.

When asking for tribute, the AI is likely (threefold increased probability) to demand the tech that it needs most, though techs that no other civ knows yet remain off-limits.	The demanded tech is chosen uniformly at random from among those techs that at least two civs know.

Rationale	Not quite the same procedure as for gifts because I wanted to keep the exception for cutting-edge tech; picking (with certainty) the most useful tech that is known to at least two civs would've been a bit awkward to implement because of the signature of the `CvPlayerAI::AI_techVal` function.
	Is the exception really a good idea? Maybe yes: The civ that receives tribute remains a threat in the medium or long term, so yielding to that civ an edge in (military) tech is rarely going to be prudent.

Gifts requests granted by human player are remembered for (on average) 150 turns.	200 turns
Rationale/ See also	For [130r](#), I've thought a bit about and experimented with the memory decay for declared wars, and ended up setting it to 150. I don't think granted gifts should be remembered longer than declared wars.
Config	`MEMORY_GIVE_HELP` (`LEADER_DEFAULTS`) in `Civ4LeaderHeadInfos.xml`.
Tbd.	Ideally, the duration would depend on the trade value of the gift, how harmful it was to the human player (tech gifts normally are pretty painless) and how badly the AI had needed the gift.

145	No diplo bonus from accepted favorite civic/ religion if no longer in that civic/ religion	
AdvCiv		BtS
The diplo bonus from having accepted an AI civ's religion or favorite civic decays much faster if the human civ isn't presently in that civic or religion, or if the AI civ is no longer in the civic or religion.		The bonus lasts for an expected 100 turns, regardless of whether the player keeps the civic/ religion. Can switch out after 5 turns.
Likewise, the penalty for refusing the fav. civic/ religion decays faster if the human civ is now in that civic/ religion, or if the AI civ has switched out of it.		
Rationale	Switching right back is a bit cheap, especially with the Spiritual trait. Until v0.95, the relations modifier was suspended instead of decaying faster, but I think that made it too unattractive to accept AI requests.	
See also	Part of the Dynamic Diplomacy changes.	

146	Hired (sponsored) war results in a peace treaty	
AdvCiv		BtS
When a civ agrees to declare war on another civ at the request of a third civ, the civ declaring war and the third civ automatically sign a 10-turn peace treaty. The peace treaty is automatically added to the trade table when a human player adds a war trade to the trade table.		No peace treaty; can ask someone to start a war, and immediately attack that someone.
Rationale	A bit of a loophole in BtS. Also want to be sure that the AI doesn't attack a human that has just paid it to attack someone else; not a problem in BtS because the AI only fights one war at a time, but could be a problem with UWAI.	

See also	Part of the <u>Dynamic Diplomacy</u> changes. Change <u>100</u> is also about sponsored wars (mostly UI changes). When there is already a peace treaty between the sponsor and the civ declaring war, <u>032</u> causes that peace treaty to be prolonged.
	<u>ctr</u> causes a peace treaty to be signed after city trades. The implementation of the UI support (trade table) for 146 is based on code written for ctr.

147	Changes to border tensions
See also	<u>004g</u> changes the explanation text from "close borders" to "shared borders". (The formula works a bit differently if <u>035</u> is enabled.)

AdvCiv	*BtS*
The number of lost tiles counted per city can be at most equal to the mean of the city's current population and its highest-ever population, or at most 6, whichever is higher.	No such per-city limit. A size-1 city enclosed by foreign borders can contribute 20 lost tiles. The number of lost tiles is then divided by the total number of owned tiles and multiplied by a personality factor to compute the diplo penalty from border tensions.

Rationale	Shouldn't get -4 relations from a single city choked city; this can easily happen in BtS, even in the late game. I generally don't like when high culture leads to painful diplo penalties.
	This change reduces border tensions a bit overall, and the following change should even that out:

When an AI civ steals tiles from a neighbor that the AI civ can't work because they're too far away from its cities, then the AI civ is slightly upset about this. The multiplier for those tiles is 5, whereas the multiplier for tiles stolen *from* the AI civ is 8. Both weights are multiplied by a percentage based on the size of the AI civ's territory and the map size; this multiplier decreases as more territory is acquired and tends to be near 50% by the end of the Classical era.	There's only a diplo penalty for stealing tiles from the AI, not vice versa. The weight for stolen tiles is 3.
The weight for the length of the border shared with another civ (no matter if tiles are stolen) is 5. The result is capped at 40, and no border tiles are counted if the border length is less than 5 tiles.	If the border is longer than 7 tiles, a flat penalty of 40 is counted. In the end, the total penalty is divided by 100 and multiplied by the AI leader's `CloseBordersAttitudeChange` value (between 1 and 4; no change).

Rationale	Count unworkable tiles because I don't want change [099b](culture decay in foreign city radii) to reduce border tensions. Should also encourage civs to put at least enough effort into culture to prevent a no man's land that no civ can work (which is implausible at peacetime). Makes sense to me that the AI is annoyed by not being able to work the tiles that it owns and is inclined to remedy this by conquering the cities near those tiles.
	All the formulas take into account the total size of the territory because, to a civ that has only 2 cities, 4 lost tiles should be a big deal, but not to a civ with 10 cities.
	I've shifted some weight from the common border penalty to stolen tiles because, when neither side loses tiles, there really shouldn't be much bad blood. The BtS formula is also too abrupt (0 or 40).
Tbd.	Perhaps let the relations penalty increase sublinearly with the weighted sum of stolen tiles. Easy enough to do in `CvPlayerAI::AI_updateCloseBorderAttitude` by e.g. exponentiating `iPercent` in the end with 0.7 and then times 3, but maybe a penalty of 1 or 2 already happens much more easily than -3 or -4 (this would be the goal).
Stolen tiles with a bonus resource are counted double.	Every stolen tile counts as 1 tile.
Tbd.	Should make the value counted per tile fractional and take into account yields as well. Perhaps there is already a suitable tile evaluation function in the `CvCityAI` or `CvPlayerAI` class.

148	Relations to attitude mapping of the AI changed	
AdvCiv		BtS
At a relations value of 4, the AI becomes Pleased, at -2 Annoyed, at -9 Furious, at 9 Friendly. Human players receive no relations penalty from any of the difficulty settings. On the lowest three difficulty settings, humans receive relations bonuses ("a first impression ...") of 3, 2 and 1 respectively.		Friendly at 10, Pleased at 3, Annoyed at -3, Furious at -10. On Noble difficulty and above, all relations suffer a penalty of -1. On Chieftain, humans get +1 relations from AI civs and on Settler +2.
Rationale	In effect, this makes Friendly 2 easier. A distance of 7 between Pleased and Friendly was too much considering that most AI civs are already very unlikely to start a war at Pleased and are willing to trade tech. The additional benefits of Friendly aren't that great. Until AdvCiv 1.0, I had kept the Friendly threshold at 10, making it (in effect) only one lower than in BtS, and I had increased the Furious threshold (in effect) by 1, setting it to -8. This way, Pleased and Annoyed attitude had intervals of equal length, as in BtS. I eventually decided that this is not an important property to uphold and that, on the contrary, a longer interval for Annoyed makes more sense because negative modifiers are easier to stack up than positive ones – mainly through declarations of war. Various changes to relations modifiers in the mod make it considerably harder than in BtS to improve relations well beyond the Pleased threshold.	
	As for flipping the sign of the difficulty-based modifier, the benefit is that the player sees the "A first impression ..." line much less in the game. It was a bit silly to tell the player that he/she is making a bad first impression everywhere.	
	The downside is that players have to relearn the thresholds that might be familiar from BULL.	

See also	About important changes that make positive relations modifiers harder to get, see the Dynamic Diplomacy chapter.
	The Show Hidden Attitude mod makes the BtS relations penalty from difficulty visible to the player (and is also included in BULL).
	There's a bullet in WIP section that links to several CFC posts (prior to AdvCiv 1.0) pointing out that Friendly relations are too difficult to attain (and too rare between AI civs).
Config	The attitude thresholds can be set in `GlobalDefines_advc.xml`.

Friendly attitude causes the AI to vote for someone in a victory vote; Pleased is not enough.	+8 relations needed for a victory vote.

Rationale	Seeing that Friendly attitude, in effect, starts at +8, this doesn't really change anything. A happy coincidence because it's much more intuitive to require Friendly relations than yet another threshold slightly lower than Friendly.
See also	115b: AI won't vote for victory when itself at victory stage 4.

AI civs have a +0.3 bonus to "first impression" relations (cf. 130b) on difficulty settings lower than Prince. From Prince to Deity, the bonus increases from 0.4 to 0.9.	No diplomatic handicaps for human players, though the AI-specific portion of the "first impression" modifier (i.e. from peace weight and warmonger respect) is more often positive than negative.

Rationale	This was a bit of an accident. Until v0.93, I had assumed that the BtS relations penalty (-1) applied only to human-AI relations, and I had meant to replace this with a +1 bonus on AI-AI relations. Now that diplomacy overall is balanced around the inter-AI bonus, I'm finding it a bit difficult to remove it entirely; I don't like when AI leaders are Annoyed at each other from the beginning.
	Relations bonuses from "fair and forthright" trade are generally easier to attain for humans than for the AI, and some relations bonuses tied to AI requests are only available to humans. This is evened out by relations penalties for rejected requests, but not entirely, one could argue. Let's say that the AI bonus makes up for these asymmetries.
See also	250d removes a few minor advantages that the AI receives through the difficulty level.

The time-based relations modifier for having the same state religion has an upper limit between 1 and 6, depending on the AI leader. I've left all limits of 3 or less unchanged and reduced all limits above 4 by one. As for leaders with a BtS limit of exactly 4, I've reduced the limit to 3 for about half of those AI leaders so that, overall, the average limit has decreased by ca. 0.5.	The limit is between 1 (Willem) and 7 (Ashoka, Zara Yaqob).
Similarly, I've reduced the limits for the time-based "wisely chosen civics" modifier by ca. 0.5 on average, reducing all limits greater than 4 by one and also almost all limits that were exactly 4, the exceptions being Lincoln (Emancipation), Mansa Musa (Free Market), Washington (Free Speech) and Willem (Free Religion).	The limit is between 1 (Saladin – Theocracy) and 6 (Gandhi – Universal Suffrage, Mao, Stalin – both State Property).

Rationale	To match the decrease of the threshold for Friendly attitude, i.e. so that the new threshold doesn't make too many AI leaders very easy to befriend through a shared religion (or civic, less of a concern). It's generally desirable to reduce modifiers that have almost no rational basis.
	For simplicity, I've changed almost exactly half of the leaders. When I had to choose among leaders with an equally high limit, I decided based on historical and gameplay considerations; e.g. Lincoln being a strong proponent of Emancipation, also internationally, is (somewhat) credible, Roosevelt (or anyone really) promoting Mercantilism much less so.
See also	Related CFC post (1st spoiler box)
	130n adjusts the time-based ideological modifiers dynamically based on their popularity.
	See the WIP section (toward the end) for a note about overhauling AI personalities for gameplay and historicity. In contrast, the changes implemented here are mostly across the board in order to make them easy to adapt to ("ideological modifiers don't go as high as in BtS anymore" – simple enough).
Tbd.	Limits for religious civics should be reduced if the state religion doesn't match. Shouldn't get a net relations bonus for sharing, say, Organized Religion with Suryavarman and running a different religion.
To become the worst enemy of an AI civ despite not being at war, the total relations value needs to be at least 1 worse than the threshold for Annoyed attitude.	Annoyed attitude is the only prerequisite.
Rationale	Going from Cautious attitude to being the worst enemy just through a 1-point decrease in relations is too abrupt and can lead to too many changes in worst enemy status, especially during the first two eras. Perhaps the worst-enemy threshold should even be 2 lower than Annoyed, but, seeing that this is a post-v1.00 change, I want to play it safe.
See also	Rank hate easily leads to Annoyed attitude in the early game. 130c may make flickering between -2 and -1 more common.
	130d changes some rules regarding vassal agreements and worst enemies; 130p prefers war enemies as worst enemies.
Config	`GlobalDefines_advc.xml`

149	Diplo modifier from resource trade	
AdvCiv		BtS

When an AI civ receives one or more resources from another civ, the resource trade counter of the AI is incremented twice per turn, each time with the same probability (cf. 130k). This probability is based on the number and utility of resources received from the other civ and the number of resources available from elsewhere (city network, other trades) in the AI civ's capital. The number of resources that the other civ has available also factors in – when it has, say, only three resources available for trade, then the recipients feel special.	The counter is incremented once (with probability 1) per turn for each received resource. It doesn't matter how many resources the AI civ already has, how useful they are or how many resources the other civ has.
The resource trade counter is decreased by 1.7% per turn when no more resources are received through trade, but can also decrease if the increment probability becomes too low.	The counter is only decreased if 0 resources are received through trade. The decrement is 1 plus one quarter of the number of cities owned by the AI civ.
The resource trade counter can't increase beyond 125.	The counter can increase indefinitely.
Resources from vassal tribute deals are counted only partially.	

Rationale	Same as 130i: Make the AI harder to please. The exponential decrease is consistent with 130k.
See also	Part of the Dynamic Diplomacy changes. 036 changes the evaluation of resource utility. 007 shows AI resource trade counters in Debug mode (Alt+Ctrl on an owned tile).

150	Additions to the replay log
See also	100 adds sponsored wars and brokered peace to the replay log. 106 excludes some things.

AdvCiv		*BtS*
150a	Switch to no state religion	
Replay and on-screen announcement indicate when a civ renounces its state religion, i.e. switches to no state religion.		Religion changes are logged and announced, as are civics changes to Free Religion.

Rationale	Can also switch to no religion without adopting Free Religion, and that's worth logging.
See also	151 below
150b	Results of diplo votes

Record successful UN and Apostolic Palace proposals including targets (e.g. the name of the city to be reassigned), the civ who made the proposal and the vote tally.	Replay only shows the vote tally for and against, without naming the proposal.

Rationale	Since I'm including the name of the Secretary-General/ AP Resident when recording proposals, there should be no need to record the Secretary-General elections.

151	Message about changed religion or civic indicates the earlier civic or religion

152	War trade (sponsored/hired war) indicators on Glance tab	
AdvCiv		*BUG*
New BUG option "War Trades" on the Advisors tab (enabled by default). If checked, the Glance tab (Foreign Advisor screen) shows a fist icon when a leader is willing to declare war on another leader.		The Glance tab shows icons for war and worst enemies. The BUG Military Advisor shows war trades (and embargo trades) on the Sit-Rep tab.
AdvCiv never shows a fist icon on the scoreboard (neither did K-Mod prior to v1.46).		The fist icon is shown on the scoreboard when an AI civ is *not* willing to start any wars.
Rationale	Don't want to use the BUG Military Advisor; too much stuff. On the Glance tab, the indicators don't require extra space because ongoing war (swords icon) and willingness to start a war (fist icon) are mutually exclusive.	
See also	210a adds an alert for war trades, but this is no use when a savegame is loaded or when the player loses track of the trades mentioned in alerts.	

153	Split human unit groups at convenient times	
AdvCiv		*BtS*
When a worker build, pillage, bombard or air bomb mission is completed by a group of (non-automated) human units, if the group has no further missions queued, any units with available moves are split into a separate group and that group is either immediately selected (if the mission has just been ordered) or included in the unit (group) cycle.		When a group of units is ordered to execute a mission, then as many units as possible execute the mission. If all units use up all their movement points this way, then (no change in AdvCiv) unit cycling continues with a different group; otherwise, the whole group remains selected.
For missions not listed above, in particular for move-to missions, groups that have available moves are included in the unit cycle even if not all units are able to move. Such groups are not split up however, and the whole group gets selected when it comes up in the cycle.		When a group completes its mission at a time when it isn't selected (multi-turn move or worker build; queued orders), then the group does not get selected through unit cycling, even if some units still have available moves.
		No change: Units in automated groups, groups with multiple queued missions and worker groups on a route-to mission will begin with the next task even if not all units in the group have available moves. However, a unit will never move to a different tile ahead of the whole group (as this would cause the group to split).
Credits	Proposed at CFC by Leoreth: post (item 1, last part of item 3)	

427

Rationale	The goal is to make sure that the player is aware of unspent movement points and to help him or her spend those points by selecting a sensible set of units. However, the game must not split up large groups that the player may want to keep intact.
	Aside: There is a distinction between selecting units (yellow frame on the UI) and forming a group. Technically, the currently selected units are stored in a CvSelectionGroup object (`CvDLLInterfaceIFaceBase::getSelectionList`; group id 0), but that group gets overwritten as soon as a different unit becomes selected. As soon as a player gives an order (skip will do) to the current selection, those units form a persistent (CvSelection-) group.
	It's preferable not to directly split up human groups and instead to just select a subset. That said, a player who doesn't want to move the selected subset (and therefore probably doesn't want to split the group), is probably going to order the selected subset to skip its turn, thereby splitting the group. So it doesn't make a big difference, and I do split them up directly because it's easier to implement.
	For build, pillage, bomb(ard), it seems fairly likely that the player will want to give orders to any units with movement points left, and those groups are unlikely to be very large or, if they are, the split is likely going to be uneven (only so many units can bombard or pillage a tile).
	For movement (including attacks), groups can be arbitrarily complex and it may well be that the player wants to keep the group together in one tile. (See also Leoreth's arguments linked above.)
	Queued missions already work pretty well in BtS; e.g. when ordering a group of workers to first build a Mine and then a Road, no worker turns can get wasted in between the two builds. If group automation and group route-to were allowed to break up the group, then the easiest implementation would be to split the group (permanently) as soon as automation or the route-to mission starts. It's easy enough for the player to do that manually for these high-level commands; and perhaps there can be reasons for wanting to keep a group of workers in a single tile.
Tbd.	Maybe there really are no good reasons for group route-to or group automation and groups should be force-split upon those commands.
See also	Smarter order of execution for group missions: 004c

When a human group of attackers (regardless of the Stack Attack option) destroys the final enemy defender, only the most recent attacker moves into the defending tile. The remaining units split up into one group with units that still have moves left and another without moves left. Unit cycling will normally continue with the former group if the most recent attacker is out of moves. When an attack into a non-adjacent tile is ordered, then this change does not prevent all selected units from moving into a tile adjacent to the enemy. From that adjacent tile, only one attacker will advance.	All selected units advance. K-Mod has fixed a BtS problem with Gunships (which can't capture cities) getting left behind after a successful group attack against a city.
Credits	Proposed at CFC by Leoreth: post (item 2)

Rationale	Often, the best play is to split the victorious stack between its original tile and the target tile so that no (potentially damaged) units are left exposed. The BtS behavior makes the Stack Attack option pretty much unusable in some situations, and, when group attacking one by one, it's also easy to miss the point when there is just one defender left and to move all remaining attackers by accident.
	Splitting units with remaining moves from units that (probably) have fought should often be a welcome convenience (though there can be cases where it's undesirable).
	Sending only one attacker forward when attacking across one or several intermediate tiles sounds reasonable, but probably would be unexpected when the attack comprises a multi-turn move. And, with the Stack Attack option, everyone will have to move. So this would get pretty inconsistent. Also seems a bit difficult to implement.
See also	004c may interrupt group missions upon air interception.

154	Unit cycling button
Credits	Discussion on CFC with mockups, screenshots. Another discussion; user Set making a case for a unit cycling button.

AdvCiv	BtS
Show a button to the left of the big flag button (i.e. at the right end of the command area) that cycles to the next group of units that needs orders. The button shows the icon of that group's head unit and a dot indicator in green if that unit still has all its moves available and yellow otherwise. Right-clicking the button cycles to the group preceding currently selected group (if any) in the cycle, i.e. normally the most recently selected group before the current selection.	Unit cycling can be forced only through the hotkeys W (I guess for "let the current selection wait"), Num5 or Enter. (Enter will instead end the turn when the End Turn button flashes red.) The backslash key (^ on German keyboards) cycles backwards.
When no group needs orders and none is selected, the unit cycling button disappears. If one or multiple units are selected and none need orders, the unit cycling button shows the icon of the head of the current selection with a white dot if that unit has moves left and with a red dot otherwise, and causes all units to become unselected when clicked (left or right).	Pressing a unit cycling hotkey will unselect all units if the unit cycle is complete.

Rationale	Experienced players need this fairly frequently, and, judging by questions posted on CFC, some are unaware of the keyboard shortcuts (the Enter shortcut seems to be less widely known than W), keep forgetting them or are even unaware after years of playing that unit cycling can be forced. Also, any player will sometimes, maybe by accident, select a unit that already has orders and whose orders they don't want to change, and, without knowing the unit cycling shortcuts, they'll be at a loss as to how to continue with the unit cycle (see this post in the thread linked above). And some few players prefer to control the game entirely through the mouse.

It's also preculiar that the game allows players to disable automatic unit cycling but doesn't provide any visible UI support for manual unit cyclic.

I don't know (yet) if showing the icon of the next unit in the cycle is really helpful, but it's an intuitive way of communicating the functionality of the button. The overlayed dots make the button look just like the unit buttons listing units in the same tile as the current selection. Also, the dot is usually green, which matches the color of the End Turn button so long as the unit cycle isn't complete. That and the placement to the left of the End Turn button communicate that cycling through units that need orders is part of the turn sequence. The city cycling arrows from BUG are also located nearby, and cities that need production orders are shown as building buttons (when playing with minimized popups) – albeit at the right edge of the screen.

On low resolutions, the command area can be filled with promotion buttons. In that case, there is still enough room for the unit cycling button in between a vertical scrollbar and the big flag. That's also where the production filter buttons are placed when the city screen is up.

I didn't want to show unit cycling as a command button ("Wait" command) because it's just the point of cycling away that it does not give the current selection any command for the time being.

The right-click behavior is a bit obscure, but, if one is aware, right-click to go backwards feels pretty natural. The other reasonable behavior I can think of (which I've tested and left as a comment in `CvDLLWidgetData.cpp`) is to focus the camera on the next group without changing the current selection. Neither of these functions are really needed, but going backwards seems a bit more useful and lets the button cover all unit cycling commands that BtS allows through hotkeys. |
Tbd.	Could show in the button's hover text how many groups still need orders. Will have to play with the button enabled for a while to figure out what extra info, if any, would be nice to have in the hover text.
Config	The button can be disabled through the BUG menu ("General" tab). In the same place, a second button for worker cycling can be enabled (see below).
See also	The deselection behavior is based on the unselect-all global control implemented by change 088.

Disabled by default: Underneath the (all-) unit cycling button, a worker cycling button is shown if there is any group that contains a Worker or Workboat unit that needs orders. Clicking the button will select the next such group in the unit cycle. The button doesn't have any right-click behavior. If both unit cycling buttons would refer to the same group, i.e. if the next unit in the overall unit cycle is also a worker, the worker cycling button is not shown.	Worker cycling is available through the slash key (# on German keyboards).

Rationale	To give the player a choice in how to proceed in the unit cycle, and so that all unit cycling commands available through shortcuts are represented by a virtual button. (There are also shortcuts for cycling through individual units in the current selection group, but that can already be emulated by clicking through the buttons in the unit list. It's also a bit of a different feature and not widely used I think.)
	Testing will show if this is really useful or if players would perhaps prefer some other choice.
	Going backwards in the worker cycle isn't implemented. Wouldn't be difficult to do, but I don't think I want to bother with this. Therefore no right-click behavior.
	Similarly, it might be nice to exclude non-worker units from the unit cycle controlled by the upper button when the worker cycling button is enabled, but I'm not going to implement that. (I also don't really think that cycling through all units is dispensable, and I wouldn't want to add a third button.)
Tbd.	Perhaps add a button for triggering auto-missions (missions of automated units and of units with queued orders). There isn't really room for such a button in addition to the two unit cycling buttons though. I guess two choices could be added to the current BUG dropdown menu: One for just the auto-missions button and one for unit cycling plus auto-missions (never mind the combination of worker cycling plus auto-missions).
	Anticipating which unit will have moves after auto-missions is too difficult to do, so the button icon shouldn't show any unit graphic. It could show a variant of the Automate button graphic with the big yellow dot in the middle edited out (because that dot represents the currently selected unit, which is not going to have an auto-missions). The hover text could show the Ctrl+A shortcut.
	Note that K-Mod triggers auto-missions when the unit cycle wraps around. I think this only happens when the last group that needs orders is either given orders or when the players tries to cycle away from that group (using a hotkey). I think BtS triggers auto-missions as soon as a player cycles manually to another unit.

155	Changes to team games
See also	Also merged some changes to team rules from Kek-Mod; see kekm.
Tbd.	Hardly tested.

AdvCiv	BtS
Trades proposed to the AI by a member of its team are considered with more goodwill than trades proposed by members of other teams, but still rejected if they're very uneven.	The AI accepts gift requests ("sure would come in handy") and gifts from team members. Can't even ask the AI to offer a trade.
The AI can reject gift requests by a teammate. It remembers granted requests and forgets them twice as fast a gift requests from non-members.	Always accepts gift requests and doesn't remember them; i.e. the human can ask as frequently as he/she likes. (Whereas the AI remembers both granted and rejected requests from non-members and refuses to consider further requests while still remembering an earlier one.)
The diplo text for a gift request to a team member is still "sure would come in handy."	

Rationale	Don't want the AI to behave like a puppet in human-AI teams. Should feel more like playing together with another human being, and these have their own opinions and tend to behave not entirely selflessly.
	Kept the special diplo text because gift requests to team members still get a special treatment (e.g. forgotten faster).

AdvCiv	Unofficial Patch
A human can't offer to an AI teammate to switch civics or religion.	Can offer this to the AI. Not sure if the AI appreciates it in any way. Probably a bug.
Rationale	The AI can't judge whether it benefits from a human civics or religion change.

AdvCiv	BtS/BUG
In team games, team membership is color-coded on the scoreboard: Civs belonging to the same team (and also their vassals) have their score shown in the text color of the (master) team leader.	BUG adds an option for grouping vassals with their masters (enabled by default in AdvCiv), but there's no way to indicate team membership on the scoreboard. Scores are always shown in white.
Rationale	Hard to tell which team is doing well when team membership isn't shown.
Config	Can be disabled through the "Scores" tab of the BUG menu (option "Color-Code Teams").
Tbd.	The colored scores can be a bit difficult to read. Perhaps Kek-Mod's "option to add team number to BUG scoreboard" is a bit better (Git commits 1 2)? It's very low-key when disabled, so I could also offer it alongside the color-coding option.
See also	004v: Other changes to scoreboard text.

AdvCiv	BtS
Teammates cause Emancipation anger when not all members of a team are in Emancipation.	Teammates are exempt from causing Emancipation anger.
Rationale	Seems a bit more intuitive.

156	Penalty for shared research goals in a team
AdvCiv	BtS
When multiple members of a team are researching the same technology, their research modifiers are decreased by 10 percentage points. The AI disregards the research goal of teammates; may or may not choose the same tech.	The research cost modifier of a team is increased by 50 percentage points per member beyond the first (still the case in AdvCiv), but whether the team members are researching the same technology isn't a factor. When an AI civ needs to select a tech to research, it always picks one that another team member is already researching.
Rationale	The BtS mechanism (teams basically always research one tech at a time) is OK, but the penalty should result in more interesting decisions.
Config	Can be tweaked or disabled in `GlobalDefines_advc.xml`.
See also	910 shows a breakdown of research modifiers in tech hover text.
AdvCiv	BtS/ K-Mod

In multiplayer games, the "you have discovered" sound is played when a tech is discovered by a teammate. Otherwise, the tech quote is played.	The tech splash screen is never shown in networked multiplayer games (still true in AdvCiv). BtS always plays the "you have discovered sound" along with the message about any newly discovered tech. K-Mod instead always plays the tech quote. (Comment in the code: "the 'MP' sound is boring.") It looks like BUG had already been playing the tech quotes (through XML changes; K-Mod through the DLL).
Rationale	Want to give the "you have discovered" sound *some* use. Should make it easier to distinguish between own discoveries and those of teammates – now that the members of a team usually research different technologies each.

157	Discourage binary research
Rationale	Binary research, i.e. setting the research slider only to extreme positions, is tedious micromanagement that should be discouraged, not rewarded.
AdvCiv	*BtS*
In the calculation of the total commerce rates of a player, when rounding errors add up to at least 1 commerce, that total error (rounded down to an integer; typically to 1) gets added to the commerce rate(s) that contribute(s) the most to the total error. Note that the rounding errors corrected this way are typically the result of splitting the base commerce rate according to the special commerce sliders, but, through science and gold buildings combined with specialists, rounding errors can even occur when running 100% research or 100% gold.	Each city counts its commerce rates, already adjusted to the slider percentages – but not yet including gold expenses, at times-100 precision, i.e. with only minor rounding errors. The total commerce rates (gold, research, espionage – total culture isn't really relevant; cf. 004p) are computed by summing up the per-city rates and dividing each rate by 100, rounding toward negative infinity. This can result in a rounding error (of usually at most 1 commerce) to the disadvantage of the player. Whether that's the case is somewhat unpredictable and will depend on the specific slider percentages. That said, in the early game, so long as there are no Libraries, it's very predictable – at 100% research or 0% research, no rounding error will occur, at all other positions of the research slider, there will normally be a rounding error. Even once there are Libraries (but no Markets), 0% research incurs no rounding error. The early game is when the rounding error hurts the most.
Rationale	For numbers that appear on the UI, I generally want to stick to the BtS rule of always rounding down. Special commerce is already treated differently in by being counted at times 100 precision, so I feel that eliminating this rounding error is fair game.
Tbd.	There are still several strong reasons for using binary research: Not committing to a tech sooner than necessary; waiting for tech diffusion multipliers to increase or for Libraries to be constructed; temporarily building up a gold reserve for trade opportunities or random events. On the other hand, partial progress toward a tech has no benefit (unless one is trying to trade for the tech).

I think there'll have to be a penalty for extreme slider positions (like in or akin to Alpha Centauri) or for adjusting sliders. Don't know how that should work in detail. |

158	Revised AI strength memory map	
AdvCiv		*K-Mod*
Moved into separate class, changed the data structure to a hash map. Enabled for human civs (i.e. for the proxy AI player running in the background) and for minor civs.		From the K-Mod changelog (v1.08): "*Added some basic AI memory so that it can remember to walk around well defended choke-points without having to cheat.*" Implemented as a `std::vector` that stores a total combat strength value for every player and for every tile. Mostly disabled for human civs "*because it may cause OOS errors*" (comment in `CvTeamAI.cpp`) and minor civs.
Rationale	Having the code in a separate class made it easier to experiment with the data structure. Normally, no strength value is remembered for the vast majority of tiles, so I suspected that a vector would lead to unnecessary CPU cache misses. In tests, a hash map was only slightly faster, so it seems that there wasn't much of a problem, but at least there appears to be a small improvement now.	
	For human civs, I see only one potential out-of-sync issue (in the cost function of the pathfinder), and karadoc had already worked around that, so disabling updates of human strength memory seems to have been an abundance-of-caution decision.	
	I don't know why it was partially disabled for minor civs; perhaps just an ad-hoc decision.	
	I've kept Barbarian strength memory enabled; will usually matter for Barbarians that emerge in the same region of the map, so it's plausible enough that they would share information. And getting Barbarians to avoid well-defended cities is a sensible goal.	
See also	[300](#) lets Barbarians use an "activity" map in addition to strength memory. The activity map steers Barbarians away from cities near which Barbarian units have recently been defeated.	

159	AI heuristic for stack power	
AdvCiv		*BtS*
Added a function `CvUnitAI::AI_currEffectiveStr` that replaces most calls to `CvUnit::currEffectiveStr`. Moved duplicate K-Mod code for counting first strikes and collateral damage into the new function. `AI_currEffectiveStr` takes the per-unit strength values to the power of 1.25.		`CvUnit::currEffectiveStrength` is equal to the unit's combat strength minus a penalty for lost hitpoints (if damaged).
		K-Mod
		The AI stack evaluation functions `CvPlayerAI::AI_localDefenceStrength` and `CvPlayerAI::AI_localAttackStrength` essentially sum up the values returned by `CvUnit::currEffectiveStrength`.
Config	The exponent is computed as 0.75 times the `POWER_CORRECTION` value set in `AI_Variables_GlobalDefines.xml`.	

See also	UWAI uses the full `POWER_CORRECTION` value. [This post](#) of mine explains why I think that a higher exponent needs to be used for strategic decisions (war planning) than for tactical decisions (imminent stack-on-stack combat).
	Old (pre-BtS) [discussion](#) about how combat strength should translate into power values. (I haven't read all that.)
Rationale	The AI had probably been overestimating the effectiveness of larger stacks of low-strength units against smaller stacks of high-strength units. Two Swordsmen don't have a 50% chance of killing a Cuirassier.
Credits	Elkad reported a game of his where the AI evacuated too readily against a stack of outdated units. [CFC post](#)
Tbd.	There may still be AI code left that estimates stack combat outcomes by comparing raw combat strength values.
	Should perhaps set a slightly smaller exponent below Prince difficulty – for that "Connecticut Yankee" feeling (men-at-arms being shredded by Gatling guns).

AdvCiv	K-Mod
When evaluating the defensive strength of a stack, the AI adds some extra value based on the sheer size of the stack.	Both the offensive and defensive evaluation sum up strength values computed for the individual units.

Rationale	It seems that the `CvPlayerAI::AI_local...Strength` functions don't take into account at all that the combat rules (strongest defender comes forward) generally favor the defender (so long as no collateral damage is involved). It's OK to underestimate the defender's advantage (let the AI err on the side of taking action), but I don't think the defender's advantage should be ignored completely, especially not for medium-size and large stacks.

160	Food after starvation

AdvCiv	BtS
After a city with a Granary loses population from starvation, the food store of the Granary is emptied into the city's food store. The Granary's capacity is only 40% of the city's capacity.	Whenever food is added to the city's store, the same amount of food is added to the Granary's store. The maximal capacity of the Granary is 50% of the city's store (whose capacity is 10 + 2 * population). After growth, the Granary's store is added to the city's store, but the store of the Granary isn't emptied. The Granary provides no food after starvation.
See also	[912d](#): The "No Slavery" option negates the capacity change.

Rationale	The Granary already provided some protection against starvation by providing a cushion after growth and helping regrow lost population. However, the cushion is just 15 food in a size-10 city. If such a city has balanced health and can't get extra food by reassigning citizens (the AI often doesn't have unworked Farms), a poisoned water supply will consume the 15 food in two turns, and then kill 6 population. My change essentially doubles the cushion, meaning that only 3 population would be killed, which seems much more reasonable. It's not a big issue for human players, but, in the Industrial era, the AI can become very vulnerable to poisoned water.
	It's strange that the Granary isn't emptied when a city grows, but normally irrelevant (it would be refilled by the time the city grows again), and actually convenient for my anti-starvation mechanism; don't need a separate food store this way.
	Since the Granary is already considered to be the most powerful building in the game, I didn't want to make it even better; and I had planned to reduce the amount of stored food at some point anyway.
Config	The capacity change is an XML change (`Civ4BuildingInfos.xml`).
Tbd.	Nerf Slavery and Drafting; this will make Granary far less powerful.

161	Probability of war trades check	
AdvCiv		*BtS*
The AI considers hiring another AI civ for war with a per-turn probability of *p* for each potential hireling computed as follows: $$p(t) = 10 / (DWTR + 10 * \min\{10,t\}),$$ where *t* is the number of turns spent at war and DWTR is `DeclareWarTradeRand` in `Civ4LeaderHeadInfos.xml`. For DWTR=40, this results in a probability of 1/4 when a war has just started (0 turns spent at war), 1/5 on the next turn etc. down to a lower bound of 1/14 after 10 turns spent at war.		`DeclareWarTradeRand` is 40 for all leaders except Pacal, whose DWTR value is 60. Regardless of DWTR, the BtS formulas result in p=1 when the war is 0 turns old. For DWTR=40, p is 1/(t+1) for the next three turns, and, for t=4 to t=10 gradually decreases to 1/40.
Rationale	I'm not sure if the guaranteed check after 0 turns was intended. If so, then perhaps the checks on subsequent turns were made probabilistic only for performance reasons – the number of (hireling, war target) pairs grows quadratically with the number of civs. That said, most of the pairs are usually eliminated by attitude checks (attitude toward the sponsor and toward the target), so I'm not too worried about performance. The BtS probability seems too high to me in the beginning and to low later on. Pacal's value may well be an accident.	

Tbd.	Considering to set leader-specific DWTR values so that some leaders tend to wait and see before bringing in help (DWTR=60), some hardly ever do (DWTR=80 or even 100) and some take all the help they can get asap (DWTR=20). Proud and miserly leaders would get a high DWTR, and profligate leaders a low one. I'm finding it difficult to fit the leaders into these categories though. Tentative: 3xDWTR=20: Huayna Capac, Louis, Justinian 11xDWTR=40: Hannibal, Suryavarman, Mansa Musa, Hatshepsut, Hammurabi, Montezuma, Saladin, Churchill, De Gaulle, Roosevelt, Augustus Caesar 14xDWTR=50: Cyrus, Brennus, Julius Caesar, Willem, Pericles, Pacal, Catherine, Washington, Joao, Boudica, Frederick, Bismarck, Wang Kon, Peter 6xDWTR=80: Alexander, Shaka, Genghis Khan, Qin Shi Huang, Gandhi, Kublai Khan, 2xDWTR=100: Tokugawa, Sitting Bull 16xDWTR=60: the rest

162	Rule changes against surprise attacks

AdvCiv	BtS
On the turn that war is declared, units that enter enemy borders have all their movement points spent. That includes units in cargo, so they can't be unloaded right away. Exception: Air units and units that can explore rival territory (Caravel, Carrack, Submarines, Spy, Great Spy, Great Merchant; notably not Stealth Destroyer).	Through cargo ships, one can attack any coastal city right after declaring war. In particular, the attacked civ doesn't get a chance to relocate civilians, missiles and ships in port, nor to engage the cargo ships (and their escort) in a naval battle.

Config	**Disabled** by default through `SPEND_ALL_MOVES_ON_INVASION` in `Civ4GlobalDefines_advc.xml`.
Rationale	See this CFC thread, and maybe also this one. I wasn't quite happy with this change from the start. A border (in the sea) is very much not an obstacle, so the loss of movement points is implausible. I also don't like that naval invasions and bombardment from coastal waters (i.e. when the two war parties share a land border) are affected by this. Fast deployment is the only reason to use navies in a land war. In testing, I also found it easy to forget about the change. It does what it was supposed to do, i.e. grant the defender an extra turn for attacks on unescorted transports, but that just comes up so rarely.

Tbd.	Have ships that unload units outside a friendly city or fort spend all movement points and maintain a connection between the ship and its cargo for another turn, depicted by the UI as some sort of overlay and labeled as "disembarking". When a disembarking ship is destroyed some or all of its cargo is destroyed as well. The idea is to expose the transported units to naval attacks for another turn without making them invulnerable to attacks by land and without slowing down unopposed landings. Will have to make sure that the AI considers naval attacks on ships that are unloading before considering land attacks on the disembarking units. Amphibic units should probably be exempt, i.e. debark instantly and perhaps even with full moves.
	Alternatively, some super-light version of zones of control might work, e.g. "Transport ships can't unload next to a hostile ship with equal or higher strength (unless they have full moves?)" or, disregarding ships in cities: "Transport ships can't unload next to a water tile with a hostile unit (unless they have full moves?)." Such a rule is more plausible than the current one based on border crossings, should be easier to remember and doesn't affect naval bombardment in land wars (nor transportation if only one side has ships). Would require some work on the AI though – escorts will have to decide whether to attack defending ships (possibly leaving the transports unguarded), transports have to decide whether to wait for a landing spot to be cleared or to find a different one, and `CvUnitAI::AI_guardCoast` should at least show an effort to protect the AI's coast against surprise landings.
	More important: Find some way to prevent surprise attacks on Workers. E.g. allow civilians (probabilistically?) to "escape" to a nearby friendly city when attacked.
See also	010 disables Worker stealing, but I don't necessarily want to keep it that way, and Workers can still be killed.
	164 makes the Blitz promotion easier to access; that promotion is helpful for destroying unescorted cargo ships before they can unload.
	098 changes the rules for culture spread across water. A similar rule change could allow borders to spread farther onto water, making it more difficult to reach the shore on the same turn as declaring war.
	Realism Invictus has a similar mechanism: CFC post (2ⁿᵈ quote box)

163	Teleported ("bumped") units have all their movement points spent	
AdvCiv		*BtS/K-Mod*
When a unit is moved because the rules don't allow it to occupy its current tile – i.e. mostly when Open Borders are canceled –, all movement points of the unit are spent, its automation is removed (e.g. if it was set to auto-explore) and it is woken up so that it'll ask for orders on the next turn. Exception: AI units that find themselves in a tile with units that they can't peacefully coexist with don't lose their movement points.		The movement points of the unit are unaffected by the forced move. K-Mod deletes all missions queued for that unit, but doesn't de-automate it and doesn't wake it up; e.g. a Scout that was fortified in foreign territory will continue to fortify after getting teleported and won't ask for new orders.

Rationale	Don't want players to use teleportation as a means of re-deploying units; can't prevent this entirely, but using up movement points is a start. It's also more plausible that even a forced move costs movement points.
	The wake-up and de-automation are just usability improvements.
	I've made an exception for the AI because, otherwise, an AI invasion could be delayed by up to two turns by placing a single unit on the tile that the AI stack will cross the border from. The teleport then pre-empts the DoW move and moves the AI stack to a non-enemy tile. Depending on the surroundings, that tile may not be adjacent to the border. Without the exception, the interfering unit could even be moved away before the AI stack gets to attack it.
See also	046 makes minor changes to the selection of the tile that a bumped unit teleports to. ctr fixes an issue with units getting bumped out spuriously after liberating a city under occupation, and wakes up human units after tile ownership changes that don't lead to bumping.

164	Changes to the Blitz promotion	
Config	`Civ4PromotionInfos.xml`	
AdvCiv		*BtS*
Then Blitz promotion requires Military Science (no change) and either Combat III or Drill I.		Military Science and Combat III. (K-Mod had allowed Drill III as an alternative requirement.)
Rationale	Want to make Blitz ships (Destroyers, Battleships, Submarines) more common because these make it easier to intercept an (unescorted) naval invasion; cf. 162.	
Tbd.	Or perhaps requiring Drill II would a good compromise.	
Units with more than 1 move and paratroopers can receive the Blitz promotion if they also have the proper combat class (Mounted, Armored, Helicopter or Naval; no change). The extra move can come from the Morale promotion.		No special clause for paratroopers, and the unit *type* needs to have more than 1 move, i.e. Morale doesn't help.
Tbd.	This still doesn't allow Paratroopers and Gunpowder units with a Great Warlord to receive Blitz because Gunpowder units aren't allowed to have Blitz. I'm reluctant to change this because Mechanized Infantry could become too similar to Tank if it's allowed to have Blitz. Not sure how to solve this problem.	
Blitz allows units to make one extra attack per turn.		The number of attacks is only limited by the number of movement points. E.g. a Destroyer could make 7 attacks per turn or (depending on promotions) even more.
Rationale	Blitz Destroyers are too goofy and also a bit too powerful; can't have that if Blitz is accessible for level-3 units.	

165	Dimensions of individual map scripts adjusted
AdvCiv	*BtS*

Adjusted the grid dimensions of Hemispheres, Not Too Big Or Small, Pangaea and Tectonics.	These maps use the default dimensions set in `Civ4WorldInfos.xml`. Their land/sea ratios are also comparable, however, e.g. Not Too Big Or Small (K-Mod's version of Big Or Small) has a much longer coastline than Pangaea and thus a higher number of resources, of decent tiles and, ultimately, of city sites.
Terra uses the dimensions only slightly larger than the new defaults (see change 137).	Terra was using the grid dimensions of the next higher map size; e.g. a Large Terra map resulted in Huge dimensions. Developer comment in `Terra.py`: "*Enlarge the grids! According to Soren, Earth-type maps are usually huge anyway.*"
See also	137 changes the default grid dimensions and player counts for each map size. The default dimensions are tailored for Fractal and Continents; therefore, these scripts don't need custom grids. 021b: Perfect Mongoose uses the default dimensions, but I've tweaked the land/sea ratio. 021a: Other small changes to the Tectonics script.
Rationale	So that the default player counts result in a about the same amount of space for peaceful expansion on all these maps. Those with uninhabited continents (e.g. Terra) should, in my opinion, have a more room once these continents become reachable, but less room initially (compared with e.g. Pangaea). I'm not sure if it's the land ratio or perhaps the resource placement, but the Old World in Terra isn't usually very crowded with the default grid size and player count. "Earth-type map = huge" may be true about scenarios (because the coastlines become unrecognizable at smaller sizes), but shouldn't matter for randomized scripts. I haven't adjusted any scripts that start with a letter before "F" because I want Fractal to be on top of the list on the Custom Game screen. The ordering can't be modded and the script names can't really be changed (see below). Instead of adjusting the grid sizes, one could add player count recommendations to the map description (in fact I had done this for a few maps until AdvCiv 0.95), but such recommendations are only shown under "Play Now", so most players never see them. As for adding a recommendation on the Custom Game screen: The displayed map name ("Terra") is the name of the map script file, and changing this file name (to e.g. "Terra - extra players recommended") would result in a second map, listed in addition to Terra. I've tried adding a dummy drop-down menu with a single choice, but apparently at least 2 choices need to be given. Such a menu would also be easy to miss because the default menus (which all maps have) are displayed first.
Config	Through copies of the respective map scripts in `AdvCiv\PublicMaps`. (My understanding is that it is the best practice to place modified BtS map scripts in the `PublicMaps` folder so that the game remembers the most recently played map even when no mod or a different mod is loaded. CFC thread on the subject; "We the People" Git issue)
Tbd.	Would actually be better to put them all in `AdvCiv\PrivateMaps` – because those maps are listed at the very top of the Custom Game screen. I don't like that the non-BtS scripts are at the top currently. However, having already gone back and forth a couple of times with the folder names, I'm not sure I should change them again; makes it more error-prone for players to update the mod.

170	(Reserved for an overhaul of religion)

170a	Effects on diplomacy
See also	[145](#) applies accepted-religion memory only so long as the religion is shared. [130n](#): religion hate based on revealed population with the offending religion [115b](#) changes how a favorite religion affects the utility of the Theocracy civic. [150a](#): religions in replays [106e](#), [151](#): notifications about religions [131](#): AI switches its state religion probabilistically

AdvCiv	*BtS*

Rationale	

171	Founding and spread of religions by the AI
Tbd.	AI shouldn't found religions before founding a second city or hitting the happiness cap in the capital. AI with a favorite religion should avoid founding earlier religions.

AdvCiv	*BtS*
When an AI leader gets to choose which religion to found ("Choose Religions" game option) and the favorite isn't available (already founded or none specified), that leader chooses the religion with the cheapest tech requirement.	Chooses uniformly at random.

Rationale	If the preference of the leader can't be matched, at least try to match the game year by picking religions in chronological order.

AdvCiv	*K-Mod*
While preparing war against a civ, AI missionaries tend to avoid spreading religions to the war target. No religion spread when war is imminent.	War plans don't affect AI religion spread.

Tbd.	Merge parts of `CvPlayerAI::AI_missionaryValue` and `CvUnitAI::AI_spreadReligion` into a single target evaluation function. Not an easy task because they implement the same ideas differently and the results aren't on the same scale.

172	Culture rate from religion unaffected by state religion
AdvCiv	*BtS*
A city's culture rate from religions is set to the maximum of the culture rates of all religions present in the city, i.e. 1 culture when there is at least one religion present, another 4 culture if the city is the holy city of at least one religion.	When there is a state religion, only the state religion generates culture (same as with happiness from religion), when there is no state religion, then the culture rates of all city religions are added up.
See also	[CFC post](#) recommending Paganism for the sake of maximizing religious culture (middle of the post)

Rationale	Reducing the culture output of a holy city to 0 when running a different state religion is counterintuitive. One can argue that a conquered holy city should not generate culture for the new owner, but BtS allows that too – the new owner just mustn't have a different state religion. Disabling the holy city culture in conquered cities would be easy enough to do, but, at least when the new owner runs the holy city religion as the state religion, holy city culture should arguably be enabled; that gets too complicated, and the Shrine culture (since it's a wonder) is already getting disabled upon conquest, which seems fair enough as a compromise.
	Another upside is that the Religion Advisor no longer needs to show culture output; makes it a bit easier to evaluate the effect of adopting a (different) state religion.

173	Changes to religion spread
See also	140 reduces the impact of the world-wrap setting on religion spread
	CFC post lamenting slow religion spread in AdvCiv 0.99, before a v1.0 tweak to the world-wrap adjustment. That player was probably mainly feeling the K-Mod adjustment of religion spread to game speed, but was perhaps still onto something.

AdvCiv	BtS/ K-Mod
So long as a particular religion hasn't spread to any cities of a player, the per-city spread probability gets adjusted to the number of cities owned by that player: more likely to spread if that player has few cities, less likely if the player has many cities.	Once a turn, each city makes a spread religion roll for each connected Holy City. K-Mod has reduced the impact of (air) distance on the spread probability. There's K-Mod code commented out (was apparently under consideration) that increases the spread probability so long as a religion has spread to at most two cities in the world; this would affect both domestic and foreign spread.

Rationale	To make foreign religion spread a bit less dependent on the recipient's city count. To make an early trade connection more rewarding and to give civs that are boxed in a better shot at receiving religion spread.
	Perhaps a somewhat needless complication; but, then, the spread formula is pretty complicated anyway.

The success probability of missionaries starts at 100% and decreases by 10 points for each religion already present in the city. This is equivalent to the K-Mod behavior for cities of size 10.	Same in BtS except that the decrement per present religion is 1/9 for rival cities and 1/11 for friendly cities. K-Mod disabled the distinction between friendly and rival cities and added a population factor that made success a bit more likely in larger cities.

Rationale	It's not intuitively clear whether large cities should be easier or more difficult to convert. Just establishing a community of indetermined size ought to be easier in a larger city, but converting a particular portion of the population ought to be easier in a smaller city. Converting large cities first tends to be the better choice anyway, so making large cities easier to convert doesn't make the gameplay more interesting.
	The K-Mod code was apparently intended to produce smaller success probabilities; the actual probabilities seem to be the result of a bug – which I haven't fixed (well, I did replace it with simpler equivalent code) because I don't think smaller success probabilities would play better. karadoc lived happily with the bug for the next 7 years (i.e. until he stopped working on the mod), so he might agree.

See also	Git commits introducing the bug: 1 2 3
	CFC post by me with screenshots of an Excel table showing the BtS and K-Mod probabilities; in a spoiler box inside the "Religions" spoiler box.

174	Divine Right research cost reduced by ca. 40%.
See also	CFC post (2nd paragraph) citing some proposals for buffing Divine Right and explaining that I don't want to make such major changes just to redeem a (very) weak tech.
Rationale	A cost change is very low-key. Will at least let whichever civ decides to go for the tech waste fewer research points. The new cost is the same as Banking and still above Paper. I don't want Divine Right to stick out as a very cheap tech.

178	Changes to Apostolic Palace (AP) and United Nations (UN) votes
See also	kekm.25 allows AP war votes against voting members. 130f allows embargo votes to sever recent deals.
Tbd.	Several gameplay changes, most importantly to the membership rules. The only thing done so far (apart from kekm.25): A temporary adjustment of the City AI evaluation of the AP as a vote source. Assigns 0 value when the state religion is shared by no other civ. Will have to replace that with code estimating the potential for spreading the AP religion after/ while building the AP. (Don't want to force an AI civ that has just founded Christianity to spread that religion around before being allowed to build the AP.) Some players really hate the AP victory. The change to the membership rules will address this.
See also	115b only counts vote source utility when aiming at a religious victory; that'll also have to change.

AdvCiv		*BtS*
UN and AP don't require Diplo victory to be enabled, but the victory votes do.		Can't build AP and UN when Diplo victory is disabled.
Rationale	The AI can handle this. Disabling Diplo victory without disabling all votes is requested on the CFC forums from time to time (two examples). Seems like players generally don't dislike voting, just the victory votes, either for reasons of game balance/ fairness (mostly AP) or because they find it irrational that the AI civs will vote for a rival's victory. That said, some non-victory AP votes appear to be problematic in multiplayer (the RtR mod disables them; see under "Miscellaneous" here); will have to disable them through XML then.	
Config	Can be reverted through `Civ4BuildingInfos.xml`. Individual votes can be disabled through `Civ4VoteInfos.xml` (though this may break savegame compatibility).	
The "None" option in the popup that asks the player to select a resolution explains that this will trigger an early election. (Unless the vote cycle is ending either way.)		Not even the Civilopedia explains this as far as I can tell.
Rationale	The early-election mechanism is OK, but needs to be communicated to the player.	

The force-peace vote requires a non-vassal full member to be at war with a non-vassal voting member.	The voting member can be a vassal and, then, the master of that vassal will be forced to make peace even if it isn't itself a voting member.
Rationale	Shouldn't compel a civ that has no vote in the matter.

179	Changes to the production ability of Apostolic Palace (AP)	
AdvCiv		*BtS*
The production bonus to religious buildings is shown in the AP's help text, Civilopedia text and actual-effects text (<Alt> key).		The only mention of the production bonus is buried in the "BtS Concepts" page about the Apostolic Palace. Actual-effects (BUG mod) shows the bonus when hovering over religious buildings but not when hovering over the AP.
The AI evaluation of the AP includes the production bonus, taking into account religious buildings that already exist and that could still be built, future religion spread and the effects on other civs and diplo relations with those civs.		The production bonus is ignored by the BtS AI. There is BBAI code that factors the bonus into the evaluation of religious buildings but not the evaluation of the AP itself.
Rationale	The ability is quite powerful; shouldn't be a secret. It's tied to the AP vote source (`Civ4VoteSourceInfos.xml`), not the building, but that's a technicality that players shouldn't have to worry about.	
See also	008a shows some of the restrictions for AP only when they apply. 008e drops the "The" from the AP's name. kekm.25 loosens the conditions for holy war votes.	
The AP grants +1 production to religious buildings, costs 350 production and allows 2 Priest specialists.		+2 production, cost 400, no Priest specialists.
Shrines allow 2 Priest specialists.		3 Priest specialists
See also	Related CFC post (2nd spoiler box)	
Rationale	+2 production is clearly too powerful, and it doesn't really help that every civ benefits. Turns Monasteries and Temples into highly desirable buildings. Those buildings are generally underpowered, but a wonder making them almost overpowered through an effect that has nothing to do with the primary functions of religious buildings (culture, happiness) is a cure worse than the disease. Also distracts completely from the more interesting AP ability – the votes.	
	Seeing that the AP has an unattractive tech requirement, I do think that it needs to be buffed a little to remain worth considering for human players. Reducing the production cost should be helpful; it's a fairly expensive wonder considering that there is no resource-based discount. But I don't see a lot of leeway for lowering the cost without making the wonder feel strangely cheap. I've considered adding another ability, one that only benefits the AP owner, e.g. +3 or +4 espionage from religious buildings, but the AP already has too much text and too much complexity. The specialists are quite low-key in terms of complexity and can actually be useful when having just founded Christianity. In most situations, they don't make the wonder much better, unfortunately.	
	I've removed a specialist from the Shrines in order to avoid making Priests (feel like they're) too abundant in the midgame. Priest slots at Shrines are mostly anti-synergetic, no real loss.	

180	Damage to cities upon conquest
See also	[139](#): AI evacuation of cities [001f](#): Cities remain revealed after conquest [023](#): Occupation countdown after conquest [099](#): Immortal culture [101](#), [099c](#): Revolt probability [116](#): AI razing [130w](#): Relations penalty for cities with foreign culture
Tbd.	Damage shouldn't be caused primarily upon conquest but during attacks, bombardment and unrest. Dawn of Civilization and Civ 4 Reimagined have already implemented such changes. Capture gold needs to be reduced; perhaps only grant it when razing. Shouldn't be possible to raze in a single turn. (Use a separate change id for that though.) Bombardment and razing should perhaps also affect war weariness.

AdvCiv		*BtS*
Defensive buildings are destroyed upon conquest even if it's a reconquest: Wall, Dunn, Castle, Citadel.		"Reconquest" means that the new owner has the highest city culture – indeed, that can't happen if the new owner hasn't owned the city at some earlier point. Defensive buildings have a `iConquestProb` of 0%, which means that they're always destroyed except upon reconquest or cession. Buildings with the `bNeverCapture` flag are destroyed even upon reconquest. (In ceded cities, even the never-capture buildings remain intact.) Only cultural buildings (Monument, Library, University, Theater, Temple, Monastery, Cathedral) have the never-capture flag.

Config	`Civ4BuildingInfos.xml`
Rationale	Since there has to be a conquest before a reconquest, `iConquestProb=0` vs. `bNeverCapture=true` really only makes a difference for buildings that the "transient" owner has constructed. Don't want Walls of that player to benefit the "new old" owner because its somewhat unexpected and because I don't want to discourage players from fortifying newly conquered cities. For unique buildings, the never-capture flag is mostly superfluous – but not entirely: when two or more players have the same civilization type, unique buildings can be captured. Don't want to destroy Bunkers and Bomb Shelters upon reconquest because those are notoriously difficult to demolish in reality. They have an `iConquestProb` of 66%.

181	*unassigned*

182	Hide tile yields of unrevealed bonus resources
See also	[003i](#): Other info leaks; see the "see also" box there.

AdvCiv	BtS
Tile yields shown in help text and via the Yield Display don't take into account resources unrevealed to the active (human) player.	Resource tile yields of owned tiles are shown based on the tech available to the tile owner.

See also	[CFC post](#) pointing out that the Yield Display gives away unrevealed resources.
	[001i](#): Other info leaks; see the "see also" box there.

Rationale	One can still spot improved Copper and Iron on flat tiles because a Mine wouldn't be allowed there otherwise. That can't be helped I think, and it's not all that difficult to figure out and, more importantly, fairly easy to spot. In contrast, the info leak for all owned tiles is easy to fix, obscure and annoying to take advantage of – will have to scan all revealed owned tiles for a production icon (so long as the owner has Iron Working and the human player doesn't – which is often the case) or, when playing without Yield Display, mouse over all revealed owned tiles. I guess an alternative solution would be to reveal all resources on owned tiles – not a serious consideration; for starters, it would be very confusing for players who are unaware of the info leak.

183	Effects of forts; rules for city defense modifiers outside of owner's borders.
See also	[124](#) deals with the effect of forts on the trade network. [121](#): worker AI changes for forts.

AdvCiv	BtS
Open Borders allow sea and air units to move through foreign forts, even if the fort owner is a rival.	Only non-rival sea and air units are allowed in, i.e. only units of the fort owner's team or that team's master or vassals.

Rationale	Streamline the rules by making forts more similar to cities. Disallowing naval movement is also inconsistent with forts allowing rival naval trade to pass through (a rule that, admittedly, rarely ever matters). Air capacity is counted per team, so that's not much of an issue (and surely no more of an issue than in cities). One problem: Unlike a city, a fort can easily be pillaged by its owner (if that were disallowed, the owner would still have to be allowed to replace the fort with another improvement). Pillaging a fort could allow a human player to trap a stack of foreign AI transports in a lake or small inland sea. (Air and sea units inside a fort get bumped out when it's pillaged; that's OK.) Easier to pull off than through canceled Open Borders – but still seems like a pretty elusive scenario.

See also	[046](#) makes teleported sea units less likely to end up in lakes. [advc.pf](#): The fort rule change has made the team-based pathfinding code a bit easier to write.
	I've replaced `CvPlot::isCity(bool,TeamTypes)` with separate functions (members of `CvTeam` and `CvUnit`) for various aspects of cities, e.g. `CvTeam::isCityDefense`. Most of those refactoring changes are tagged with just "advc".

Tbd.	If AI units trapped in a lake ever become a problem in an actual game, perhaps I can get the AI to unload its transports.
	The rules for fighting within the borders of a third party (that is a rival of both war parties) could use work. Basically such fighting shouldn't be allowed; not plausible.

Can only rebase aircraft to a revealed fort or city. When a fort gets replaced or destroyed, it is removed from all players' maps regardless of fog of war.	Can find forts and cities in the fog of war by hovering with the cursor in Rebase mode. Other than that, forts are subject to fog-of-war rules (like all other terrain improvements).

Also address some other info leaks, e.g. the Fortify action recommendation.	
Rationale	These info leaks get more problematic when Open Borders allow aircraft to enter rival forts. With the info leaks closed, destroyed forts can't be secret – otherwise aircraft would be able to move to tiles where a fort had once existed.
Units in an unowned fort tile benefit from all defensive advantages of forts.	Unowned forts grant +25% tile defense; the other defensive advantages are tied to the acts-as-city ability and apply only to non-rival units: City defense modifiers (e.g. Archer; also: City Raider promotion on the flipside), immune to Flank Attack, immune to targeted attack (Ballista Elephant).
Rationale	Counterintuitive that the 25% apply but not the other defensive abilities. Arguably a relic from Warlords when the defensive modifier was the only ability. Workers being allowed to build forts on unowned tiles also seems like a bit of a relic. If the defense modifier on unowned tiles were taken away, there would be no point in building unowned forts at all. Through city razing, forts could still end up without an owner though, and then it would be rather strange that units can't get any use out of an abandoned fort.
Tbd.	Disallow forts building on unowned tiles? Then players wouldn't have to wonder whether such forts can function as ports or air base (they cannot). Forts that end up unowned through razing could still grant defensive benefits.
See also	[119](#) prevents chopping on unowned tiles.
Units defending in cities or forts of a third party are treated as city defenders only if that third party is also at war with the attacker – Open Borders alone aren't enough.	Units defending in proper cities of a third party – but not in forts – are treated as city defenders.
Rationale	To resolve the inconsistency between cities and forts, and part of an effort to make warfare in third-party territory less unrealistic. In particular, the culture defense modifier represents (in my book) the support from the local populace and a boost to morale for defending people's homes – not at all appropriate when the attacker isn't at war with the city owner.
Tbd.	Ultimately, I think I'd like to disallow attacks in the territory of a neutral third party entirely. Ordering such an attack should bring up a popup asking to declare war on the third party.

184	Military happiness only from units of the same team or vassal/master	
AdvCiv		*BtS*
A city that fears for its safety or that is under Hereditary Rule can receive happiness from military units only if those units belong to the same team as the city or to a vassal or master of that team. Only units of the same team as the city owner count toward culture garrison strength.		Any (non-hostile) units cause military happiness and count as culture garrisons; even units owned by the player that is trying to flip a city can count as culture garrison against the flip chance.
Rationale	Comes up rarely, but highly counterintuitive when it does. Arguably an oversight (already in Vanilla Civ 4).	

185	Changes for mods with modified era progression; tagged with "advc.erai" in the code.
AdvCiv	*BtS/ BBAI/ K-Mod*
Changed some AI code that made use of absolute era numbers or era differences in order to make it work better for mod-mods with more or fewer eras than BtS/AdvCiv.	Era numbers (0 for Ancient, 6 for Future) are used pretty commonly in AI code as a coarse measure of the game progress or the technological progress of a particular player or team. Usually, AI modders have used ratios involving the current era number and the total number of eras. Such code can be expected to work (somewhat) well in a mod-mod with a different total number of eras. In some cases, however, only the current era number is used, or a difference between the current era number and some threshold or target value.

Rationale	I'm aware of only one AdvCiv mod-mod with a different era count – "These Fractured States of America" ([CFC thread](#)), and development didn't get very far on that. There are plans for a merge of Greek World with AdvCiv ([CFC post](#)). So these AI changes might matter; they should have no effect in AdvCiv itself.
Config	Some of the AI code (in the DLL) relies on era ratios. Most of that code is scattered, so mod-modders can't easily change anything about it (the new function `CvEraInfo::normalizeEraNum` covers only a few cases). Hopefully no changes are necessary. Other AI code uses functions `AI_getCurrEra` to map era numbers encountered in a game to the BtS era numbers (between 0 and 6) that the bulk of the AI code was written for. All the `AI_getCurrEra` functions ultimately call `CvPlayerAI::AI_getCurrEraFactor`. However, it shouldn't be necessary to modify that function either because it uses era numbers set in `Civ4EraInfos.xml`. That XML file also designates some special eras, for example the era (if any) during which the AI can expect oceangoing transports to become available. Mod-modders should review those XML settings. They're documented in `Civ4GameInfoSchema.xml`. (Setting those special eras in `GlobalDefines.xml` would've been a bit cleaner, but I wanted to put all the era-related AI customizations in a single place.)
Credits	In part based on [this](#) Kek-Mod Git commit. (A small part of that commit has been adopted directly under id [kekm.36](#).) I've revised a lot more code than DarkLunaPhantom, but that may have been mainly older AdvCiv code. About half of the BtS/K-Mod code modified by Kek-Mod I've left alone. Some of it merely replaces code that was already based on the total number of eras with a call to a normalization function. That's cleaner I guess, but usually not quite equivalent and, to me, not worth the risk of breaking something. In other cases, I think it's better to keep using absolute era numbers. Hard to say sometimes. I expect that a mod with fewer eras is still going to give players enough time to settle the whole world, so the eras are going to last longer than the BtS eras. The mod could achieve that through additional techs – or through higher tech costs. On the bottom line, I would expect some statistics, e.g. tile yields, to develop within a more narrow span than in BtS when there are fewer eras.

186	City bar hover text
See also	[189](#): Starvation turns on city bar [188](#): BUG anger timer option also affects city bar hover text. [187](#), [101](#): Options for showing revolt help, air unit capacity in city bar hover text.

	002f: Changes to city billboard icons
	095: Option for wide city bars (only the first half will trigger hover text however)
Credits	Based on BULL

AdvCiv	BULL
Adopted everything from BULL except the "Base Values" and "Base Production" info. No options except for the building list (icons/ names/ disabled) and those listed above under "see also." Endeavored to make the formatting more compact and more self-explanatory. Minor innovations: Larger specialist icons; buildings sorted alphabetically; showing output of production processes (e.g. Wealth) and showing stored production (chopping, overflow).	Options for everything: health, happiness, hurry/ draft help, trade routes, commerce yield, culture turns, GP turns, specialist display, building list.

See also	Requested only once I think: CFC post (2nd bullet)
Rationale	Mostly for completeness' sake; to bring AdvCiv closer to having strictly better UI support nthan BULL. And I guess players used to consulting the city bar hover text would miss the BULL improvements sorely – the BtS text misses some obvious conveniences.
	The Base Values/ production stuff is too obscure in my opinion.
186b	"Zoom City Details" merged from BULL

AdvCiv	K-Mod/ BULL
The city bar help is shown when hovering over the city details button of a choose-production popup or a city name on the Domestic Advisor. Not optional.	K-Mod had not merged this from BULL. In BULL, the help text on the Domestic Advisor is shown when hovering over the examine-city button and there is an option to disable it.

Rationale	Hover text on the Domestic Advisor gets left-aligned and then obscures then examine-city button and part of the city name. That's pretty annoying when one is trying to click on the examine-city button – will have to hit it blindly through the hover text overlay. This is much less of a problem when attaching the hover text to the city name. May still take a little bit of getting used to because the cursor will usually be moved across the city name before clicking on an examine-city button. Hopefully, an option will not be needed this way. Having the city bar help for every city in one place should be a significant upside.

187	Show air unit capacity help in tile hover text

AdvCiv	BtS
Help text about the available capacity for air units is shown in tile hover text.	Shown in city bar hover text.

Config	BUG option (Map tab) for restoring the BtS behavior.
Rationale	Info that concerns both a city and the units stationed there is more convenient in tile hover text (where the units are listed). Also, capacity info can be shown for forts this way. That said, tile hover text can be scarce due to unit stack sizes. Therefore, I'm keeping the BtS behavior as an option.
See also	Uses a smaller version (75% game font file) of the BULL airport icon added by 002f.

[101](#) adds an option for showing revolt help either in the city tile hover text or in the city bar hover text. Another example of help text that concerns both units (culture garrisons) and a city. [061](#) makes unit help (in the tile hover text) more compact.	

188	Tweaks to BUG anger timer	
AdvCiv		*BUG*
The anger timer also covers anger from defying a diplo resolution (if that's the longest anger timer).		The anger timer next to the happiness display (city screen) shows the maximum of the current hurry anger and draft anger.
Hover text of the happiness display shows the remaining duration of temporary anger in parentheses.		No change to the BtS hover text.
The option also enables anger timers in city bar hover text.		Separate option for city bar hover text.
See also	[186](#) covers city bar hover text (including anger timers).	
Rationale	Defiance anger isn't very likely to overlap with other types of anger; seems like a nice little helpful addition. Showing the remaining durations in hover text is helpful when hurry, draft or defiance anger timers overlap, and – more importantly to me – helps explain the meaning of the anger timer shown (as a naked number) next to the happiness display.	

189	Starvation turns on city bar	
AdvCiv		*BtS*
Show turns until starvation on city bar.		Only shows turns until growth. BULL has a "food assist" option that shows starvation turns (but only in hover text).
See also	[186](#): Other (BULL-based) changes to city bar hover text.	

190	Additions to the Settings tab (Victory screen)
See also	[106h](#) shows part of the Settings info in the first replay message. [251](#) may show the game start turn. [135d](#) may hide the game end turn in multiplayer. K-Mod shows the game difficulty in multiplayer (in addition to the player difficulty). [300](#) used to show the earliest turn on which Barbarians can appear.
190a	Show mod name, disabled victory conditions
AdvCiv	*BtS*

List any disabled victory conditions in the game options column.		Victory conditions only covered by the Victory tab – which doesn't explicitly say which victory conditions are disabled.
Show the mod name.		The mod name is shown on the main menu screen; from within a game, it's not visible. (Some mods show it when hovering over the main screen flag button.)
190b	Show custom map options	
The values of custom map options are listed on the Settings tab (Victory screen) if they differ from the default. For example, normally, "Cylindrical" (world wrap) is not listed because that's the default value, but "Toroidal" gets listed.		BtS doesn't show custom map options anywhere once the game has started. No way to tell what the options were after loading a savegame.
190c	More info about unmet players	
Show the initial player count and whether civs and leaders have been assigned randomly. For players whose civ or leader has been set manually (as opposed to randomly), show that civ and leader information even for players that haven't yet been met.		Once a game has started (or after loading a savegame), the number of players that haven't yet been met can only be determined by revealing the map. (Well, there are some information leaks in BtS; most of which should be fixed in AdvCiv; cf. 077 – Demographics tab.)
Tbd.	When playing with the R&F option, the RiseFall component should keep track of players met by previously active players and show the civs and leaders of those players on the Settings tab even if they were randomly chosen.	
Ignore BUG options for showing civs or leaders on the scoreboard if that info would give away the Random civ or leader of an unmet human rival. Don't use player colors either.		In multiplayer games, human rivals are shown on the scoreboard even if they haven't been met. The color of the display name gives away the civ of the rival even if that rival is playing a random civ. BUG options for showing the leader name will also give away the leader.
		K-Mod already ignores the BUG option that gives away the leader name (regardless of whether the leader was already revealed on the Staging Room screen); doesn't protect the civ name.
Rationale	Seems more interesting to keep this information concealed. Provides an incentive for choosing random civs and leaders, but, in a competitive match, it is probably still far better to pick a strong leader by hand. I also don't think it's vital to know that a human rival has e.g. randomly received Tokugawa. So I don't see a balance or fairness problem with this, and K-Mod had already gone down this road a bit.	

191	Changes to the selection of random civs	
When both the civ and leader of a player are set to "Random" during game setup and the Unrestricted Leaders option isn't checked, then the selection of the civ is biased toward civs with multiple available leaders. Such civs are given a 20% greater weight for each leader beyond the first than civs with only a single available leader.		The randomization procedure is hidden away in the EXE. Judging from calls to CvRandom in the DLL, the EXE apparently first chooses random civs and then, in a second step, random leaders to match the (randomly or manually) chosen civs.

Config	`PER_EXTRA_LEADER_CIV_SELECTION_WEIGHT` in `GlobalDefines_advc.xml`.
Rationale	Leaders of the same civ have similar personalities for the most part, so, for maximizing variation in AI behavior, favoring civs with multiple leaders isn't all that helpful. The main advantage is arguably that humans will receive a greater variety of traits. (One could argue that repetitive human traits can be prevented by restarting. I think many players don't like restarting.) The main argument I see against a bias, apart from simplicity, is that a uniform selection among civs leads to the greatest variation in unique units and buildings. This is also an argument against using a greater bias when selecting human civs than for AI civs.
See also	Implementation based on [190c](#)
Tbd.	BUG option to `reRandomizeCivsAndLeaders` (CvInitCore) when regenerating the map. Can use `CvPlayer::changeCiv`, `changeLeader` for that. However, the current randomization code only handles players that have both their civ and leader set to "Random". It also bails on various edge-case conditions. Synchronization: CvGame should pass a random number from `CvGame::getSRand` to `reRandomizeCivsAndLeaders` to avoid including `CvGame.h`. in `CvInitCore.cpp`. The AdvCiv randomization code also bails when a mod-mod contains any leaders playable only by the AI or only by humans. This was somehow (don't remember how) awkward to implement.

192	Smarter city AI choices regarding first border expansion	
AdvCiv		*K-Mod*
City AI prioritizes border expansion more when there are valuable bonus resources in the outer ring. And made a K-Mod limit on the construction turns of culture buildings less strict (taking into account that the citizen assignment may change once the city starts constructing a culture building) and fixed an issue with the city's overall culture weight (typically low in the early game) decreasing the utility value counted for being able to mwork additional tiles.		Not sure if, prior to K-Mod, there was specific AI code at all for constructing culture buildings for the sake of border expansion. The K-Mod code simply counts a constant utility value and adds a shortcut for culture buildings that are quick to construct (at most 10 turns on Normal speed).
Rationale	Can be excruciating to watch the AI not claim important strategic resources for dozens of turns.	
Tbd.	The code is still pretty crude and still doesn't expand borders as early as I'd like. Could do a pretty precise calculation of whether the outer ring would get worked and whether workers are available for improving those tiles. But the bigger issue is that the K-Mod shortcut for culture buildings only looks at the culture building with the highest overall utility value. That is often going to be a Library, but Library will then often (rightly) be dismissed for taking too many turns. I've only worked around this problem to an extent.	
See also	[303](#): Barbarian cities don't endeavor to expand their borders.	

193-199	*unassigned*

200	Reverted K-Mod changes – see the chapter on [K-Mod](#).
See also	Some of the K-Mod changes are explained in the changelog. In some cases, further rationales can be found on K-Mod's GitHub page, e.g. regarding corporation maintenance: [Git commit](#)
	Prior to AdvCiv 0.98, I had restored the BtS tech requirements (Mass Media + Corporation) for Civilized Jewelers. Now the K-Mod requirements apply again (just Corporation). See CFC discussion [here](#). The essential arguments to me are that, at Mass Media, a Great Artist will virtually always do more good through a Great Work than by founding Jewelers, and that Mass Media already had two powerful culture effects prior to the BtS expansion – outweighing the (valid) counterarguments that Jewelers, being annoying in the hands of the AI (corporation spread, culture spread) and having only narrow uses for humans, shouldn't be the easiest corporation to access, and that Mass Media is a historically plausible requirement. While it is now the easiest corporation to access, it is still not exactly easy to access, especially for the AI, because a particular Great Person (Artist) is required. Insofar, an early Industrial extra tech requirement representing machine shops wouldn't make much of a difference. Requiring a Merchant would make the corporation far less useful for a culture victory without necessarily making the AI less likely to stumble onto it.

201	Tweaks to building and corporation culture rates
Reverted most of the K-Mod changes to culture rates of buildings (Cathedral, Monastery, Madrassa, Pyramids, Hanging Gardens, Colossus, Chichen Itza, Great Library, Angkor Wat, Hagia Sophia, Spiral Minaret, University of Sankore, Shwedagon Paya, Notre Dame, Taj Mahal, Kremlin, Creative Constructions – all marked with "advc.200" in XML), set a culture rate in between the BtS and the K-Mod value for Stonehenge and Temple of Artemis, and kept the K-Mod culture changes to Oracle, Great Lighthouse, Parthenon, Statue of Zeus, Mausoleum of Maussolos, Sistine Chapel, Academy, Sid's Sushi and Civilized Jewelers.	
Config	These are all XML changes (`Civ4BuildingInfos.xml`, `Civ4CorporationInfos.xml`).
Rationale	The wonder culture changes aren't a big deal, but they're quite visible and some players may actually have some of the wonder culture rates memorized. This really shouldn't be necessary; should be possible to balance things under the hood (which also helps rein in culture spread from non-building sources). Well – one could of course take the total culture rate of a city to, say, the power of 0.8 and times a normalization factor, however, the K-Mod culture spread formula is actually fairly elegant and almost simple; I'd rather not use such a crude device. Instead, I'm leaving it at a small tweak to the "free" base culture rate along with my culture decay mechanism (see under "see also"). This probably doesn't fully offset the K-Mod changes to culture spread, so ...
	I don't think karadoc has stated a rationale for his building culture changes; the changelog does say, specifically about the religion culture change to Sistine Chapel "*culture is more powerful now; so this building needed to be nerfed.*" I see a problem mainly with wonders dominating border disputes in the early game. Since tile culture accumulates turn by turn, I don't think any midgame and late-game culture sources are problematic – unless they work as multipliers. So I'm leaving the K-Mod changes to Free Speech and Sistine Chapel in place, and partly the changes to Ancient and Classical wonders, also taking into account the overall power level of the respective wonders.
	(Midgame wonders not having much impact on borders and, for that matter, Culture victory, is not a good state of affairs. However, substantially increasing post-Classical wonder culture rates would be a bit outside of the scope of AdvCiv – and should be

	accompanied by a change to the 1000-year culture doubling rule.)
	I had been meaning to leave the change to Cathedral (only 40% culture rate, only 240 cost) in place because making Culture victory less dependent on religion seemed like a noble goal. However, I haven't found enough decent measures to balance out all the ways in which K-Mod hampers Culture victory. I also think that taking 10 percentage points away merely inches away from religion as the central element of Culture victories, and a 40% modifier isn't used anywhere else so far.
See also	908b decreases the culture rate of the Creative trait and the Incan Terrace. 201 increases the culture rate of the Great Wall. 251: Removes the culture rate modifier that K-Mod adds to Colosseum. And is concerned with the balancing Culture and Space victory. 099b: Decay of tile culture, especially on "stolen" workable tiles. 098 increases the base rate of culture spread from cities a little bit. (And changes the culture doubling rule, but I've reverted that change again, leaving it commented out in the code.) 172 makes culture from city religions non-cumulative.

Increase the culture rate of Cristo Redentor from 5 to 6.	
Rationale	To match Statue of Liberty. I don't think players know culture rates of late-game wonders by heart, so making things a bit less arbitrary should have no downside.
National Park, Red Cross and all Corporate Headquarters have 0 culture rate instead of 2 or (National Park) 3.	
Rationale	Streamlining. Those culture rates are negligible in the late game. I guess they were implemented with a rule in mind that all national wonders produce a bit of culture. I don't think players will notice or mind if that rule isn't followed in the late game.

210	Additional Civ4lerts
See also	135b makes `MoreCiv4lerts` work in Hotseat 106c prevents Civ4lerts from triggering after loading a savegame 106d changes the default settings for BUG alerts and disables some alert triggers. 127 suspends alert updates during AI Auto Play 071 shows a message when first meeting a rival; configurable on the "Alerts" tab. ctr adds a city trade alert
I'm not adding a second Alert tab to the BUG menu for the new alerts – the BUG menu is already huge. Instead, I'm removing BUG alerts to make room; some aren't really useful in a mod like AdvCiv, others are made obsolete by the new alerts.	
Disabled although the space isn't needed yet: "peace treaty", "pending border expansion". I don't see how the latter could be useful, and the former is subsumed by the "willing to talk" alert. And I've made the Reminder options a bit more compact.	
Tbd.	Get rid of the "willing to sign Open Borders" and voluntary vassal alerts. Instead simply have the AI contact humans right away when it becomes willing (the turn on which a human discovers Writing will need special treatment). Merge the Permanent Alliance, voluntary vassal and capitulation alerts in order to make more room. "<team1> has signed|canceled Open Borders with <team2>." "You have gained|lost access to a source of <bonus> (now <n> sources)." "The peace treaty between <team1> and <team2> has ended." "<team1> is willing to make peace with <team2>."

	"Can now|no longer access demographics about <civ>."
	"<team1> has discovered <tech> [(trade from <team2>)]. [Obsoleted wonders: …]" (Try merging this with the tech trade alert. Should then also stop reporting "will no longer trade" and add a willing-to-talk check like the one for city trades [ctr].)
	"<team1> is willing to stop trading with <team2>." (But what about embargoes that the AI would agree to if the player didn't have any deals with the target?)
	"5 turns have passed since the last revolution; it's again possible to change civics."
	"<leader> can be convinced to convert to <religion>."
	"The enemy has been spotted near <city>." (Currently, the game reports the same enemy time and again; should work better as a Civ4lert that remembers which enemy units had already been visible on the previous turn. Sentry [004l] already works this way.)
	"Temporary [anger_icon] has subsided in <city>." (idea by CFC user BaneFire)
	"The borders of the <civ> Empire have expanded./ Tiles have flipped from <civ1> to <civ2> control." (With an indicator bubble. Covering any tile ownership change not covered by city border expansion and city-founded announcements.)
	and possibly: "A tile near <city1> has flipped to <city2>." (How to describe the tile?) "<leader1> is now [e.g.]furious toward y (was annoyed)." "Can cancel this deal with <leader>: you're receiving <item list1> for <item list2>." "<leader> has constructed a <building>|the <national wonder> in <city>." (cf. 045) "You|<leader> have|has overtaken <leader>|you in military power." "The military power of <leader> has increased substantially over the last 10 turns." (probably too noisy) "The effect of our counterespionage mission against <leader> has ended." (A message about enemy counterespionage ending would be difficult because we only learn that "enemy spies appear to be extra vigilant to our espionage efforts," but not whose spies, and the durations can overlap.) "<leader> has acquired the <wonder> of <city>." (The *Tbd.* under 106 would be preferable, but an alert that is checked at the end of turn would be easier to implement.)
	Should store AdvCiv4lerts data in savegames. Not challenging to do; just add virtual `read` and `write` functions, call them from CvPlayer and disable the `check` calls from Python upon loading a savegame. Actually, it would be best to move the implementation entirely into the DLL; there's no reason for involving Python. Moving the original (BUG) Civ4lerts into the DLL would be a much taller order. Don't need to do that though; I don't think storing the AdvCiv4lerts data in savegames will make any observable difference, so there won't be an apparent inconsistency with the BUG alerts.
210a	War trades (= hired war/sponsored war/joint war)
See also	The RevolutionDCM mod also has such an alert, but I didn't know this when I implemented the alert for AdvCiv.
	UWAI uses a different procedure than BtS/ K-Mod to decide whether to entertain joint-war offers (but this alert works in any case).
	152 shows currently offered war trades on the Glance tab.

AdvCiv	BUG/ K-Mod
Triggers when an AI civ becomes willing to	No alert about this, but BUG shows a fist icon on

| declare war on another civ at the player's request.

Shown in the "Trade" column of the BUG Alerts tab; enabled by default.

Removed the "Victory" alerts to make room.	the scoreboard when a civ is unwilling to declare war on account of "having too much on [their] hands." Prior to v1.46, K-Mod did not show this icon (not even as an option). K-Mod 1.46 reverts to the BUG behavior (icon enabled by default).
Rationale	The K-Mod 1.45 approach (no UI support for joint wars) certainly wasn't ideal, and bothered some players greatly. With UWAI, "too much on ours hands" doesn#t say much, so a scoreboard icon or alert specifically for that isn't useful. The victory alerts seem useful only in HoF games.
Config	Can also show an alert when the AI stops being willing to declare war: `ALERT_ON_NO_LONGER_WAR_TRADE` in `GlobalDefines_advc.xml`. The fist icon can still be added to the scoreboard string (letter 'M') and will work as in K-Mod if UWAI is disabled – with one small modification: No indicator is shown next to civs that are already at war; it's obvious that they have "enough on their hands."
UWAI rejects war trades when already in a war and the new target is not at war with the sponsor and farther away than the closest current war enemy: "We have enough on our hands right now."	AI refuses war trades when it is in any war, regardless of other circumstances. (This is still the case when UWAI is disabled.)
Rationale	Need to reduce the amount of messages produced by this alert. War against an additional target rarely makes sense; the hired AI civ isn't going to send units.
If UWAI is disabled (legacy AI option), then an alert is also shown when the AI will respond "we have enough on our hands" to any war trade and when the AI no longer gives such a response.	No alert; just the scoreboard indicator.
Rationale	An alert seems more convenient; this way one doesn't have to keep an eye on the scoreboard. The scoreboard indicator may still be useful in addition so that players don't have to remember which rivals are currently preparing for war.
210b	Revolts
Triggers when revolt probability changes from 0 to greater than 0 in a city, or when occupation (from conquest or revolt) ends in a city unless the city also needs orders.

Shown in the "City" column, taking the spot if the BUG "Pacification/ Pending" alert.

Until v0.94, the alert also triggered when the revolt chance became 0. | The only way to learn about changes in revolt probability is to check the city screen each turn.

The Pacification alert triggers when occupation ends in a city; the pending version when occupation is about to end. Without the alert, there is no notification about ending occupation, although a conquered city will ask for production orders when occupation ends (unless the new owner has already put something in the queue). |
Rationale	Now that revolts play a more prominent role (see 099c), an alert is direly needed. The Pacification alert seems fairly unimportant to me, but I've still included it with the new alert. Though not the "pending" option; I don't see why one would want that. No pacification alert when the city also needs orders; seems superfluous then.
210c	City founded alert removed
When a foreign city is founded in a tile that is revealed to the player, the game shows a message about this to the player. No option to disable this. No messages about cities founded	The description says that the alert is triggered when "a rival founds a city," but I think it also works for vassal cities (which aren't rivals). K-Mod prevented the alert from showing cities

in unrevealed tiles.	founded in unrevealed tiles and set it to disabled by default.
Rationale	AdvCiv sticks with the K-Mod principle that unrevealed cities should be secret. No need for an optional alert then. Frees up space for a new alert (see *Tbd*. under 210).
See also	[106](): City-founded messages in replays [ctr](): City trade alert messages

210d	Third-party resource trades
Triggers when a civ starts or stops trading a strategic resource to another civ, and when a civ starts trading any resource to a civ that it was previously not trading any resources to, or when the only resource trade between two civs is canceled.	Resource trades are public knowledge and can be looked up on the Foreign Advisor screen. The BUG Resource Trade alert triggers when an AI civ becomes willing to trade a resource to the player or stops being willing ([106]() disables the latter part).
Rationale	Was helpful for testing [036](). Triggers way too often when all started and ended resource trades are reported. Reporting only strategic resources and the first and last trade is OK, but not quite as interesting as I thought it would be.
Config	**Disabled** by default. Can be enabled from the BUG menu, Alerts tab, column Trading. There's also a debug mode for reporting all resource trades, but that can only be enabled in the DLL (`AdvCiv4lerts.cpp`).
Tbd.	Perhaps this alert should only cover strategic resources; the current scope is a bit complicated. A "third party" checkbox for an alert about inter-AI tech trades (cf. *Tbd.* under 210) could be placed in the same column of the BUG menu.

210e	AI willing to import from human
The resource trade alert (enabled by default) triggers also when an AI civ becomes willing to import a surplus resource from the player at a price of at least 3 gold per turn. It doesn't trigger when the resource is consumed by one of the player's corporations, nor when an AI civ stops being willing to import a resource.	The alert only checks AI willingness to export resources.
See also	Such an alert wasn't necessary prior to [036]() because the AI was always willing to import human surplus resources. [106](): The alert doesn't trigger when the AI stops being willing to export a resource.

250	Changes to handicaps
See also	[104p]() sets the target size for AI invasion stacks based on difficulty. [126]() deals with free AI techs when starting in a later era. [108b]() increases the bias toward better starting locations for the AI. [313]() gives human units an anti-Barbarian bonus on Monarch, Emperor and Immortal, and reduces human and AI bonuses against animals. [148]() Makes inter-AI relations improve slightly with each difficulty level.
250a	King handicap
New difficulty setting "King" with big initial AI	Initial AI advantage and ongoing advantages go

advantage (free Worker, free Pottery, humans get the worst starting plots), and moderate ongoing AI advantages (akin to Monarch).	hand in hand for the 9 BtS difficulty settings.
Uncoupled difficulty rating from handicap id. A difficulty rating from 0 to 100 is now assigned explicitly through a new XML tag. Settler to Immortal at difficulty 0, 10, 20, ..., 70; Deity at 85 and King at 65. Plus another 30 when playing a One City Challenge.	Handicap ids from 0 (Settler) to 8 (Deity) are used as a measure of difficulty, e.g. when computing player scores at the end of a game. This method breaks when handicap settings are added. If added at the end of `Civ4Handicap Info.xml`, it's treated as more difficult than Deity; if added in the middle, all the handicaps below increase in difficulty.
Rationale	for Pottery: Free tech from the first row (e.g. Hunting or Agriculture) puts those AI civs at a disadvantage that always start with these techs. Pottery should also put the AI on a path to long-term development (Granaries, Cottages), rather than using its initial advantage to choke the human players.
Config	I've removed this difficulty level in v0.90. Left is commented out in `Civ4HandicapInfos.xml` at first, deleted that too in v0.98. Monarch combined with SPaH has a pretty similar result, and I wanted to reduce the complexity of the AdvCiv mod, at least on the surface. (An additional difficulty level is a very visible change from BtS.)
Tbd.	The victory score should take into account other game settings (through difficulty or directly), e.g. Always War.
See also	108 allows the game to give the worst starting plot to a human civ. kekm.22 rounds the difficulty setting in multiplayer to the integer nearest to the average. 126 prevents civs from continuing research paths that start at free tech whose requirements aren't met (e.g. can't go from free Pottery to Archery when missing the Wheel).

250b	SPaH; see also chapter Start Points as Handicap.
Config in scenarios	Since v3.17 (reportedly), the Base Points box is grayed out on the Custom Scenario screen unless Advanced Start is set as a default option by the scenario. As a consequence, Advanced Start and SPaH (which builds on Advanced Start) are unavailable in most scenarios. I haven't found a way to work around this. However, it's easy to add Advanced Start as a default option in a scenario file: Locate the file in `Sid Meier's Civilization 4\Beyond the Sword\PublicMaps` (CivBeyondSwordWBSaves) or `Sid Meier's Civilization 4\PublicMaps` (Civ4WorldBuilderSaves). This also works for `EuroWorld.CivBeyondSwordWBSave`. Open it file in a text editor. (Or better make a backup copy first.) Somewhere between `BeginGame` and `EndGame`, add this line: `Option=GAMEOPTION_ADVANCED_START`
See also	Chapter Start Points as Handicap Advanced Rivals mod, precursor of Advanced Civ

Tbd.	Allow players to enter arbitrary point distributions, in particular ones that assign start points to human players. Obstacles: • Can't add a drop-down menu for this to the Custom Game screen; if it's done in-game, it will have to happen after game start. For multiplayer, it's not clear who should enter the points; and they'd have to be explicitly synchronized. • I've tried an XML file, but that doesn't work well for multiplayer – not clear to the players if they should all use the same XML file, or if XML data of the host will be synchronized with the others. I'd rather avoid using XML for custom game settings. Caveat: Code added in `CvPlayer.cpp` assumes that only AI civs can have Advanced Start when SPaH is enabled.

The new SPaH option takes the spot of No City Razing (NCR) on the Custom Game screen; NCR moves to the bottom.

SPaH can't be used in scenarios because the Base Points box isn't available on the Custom Scenario screen.

See also	[250c](#) changes the default start points on the Custom Game screen, and increases the start turn based on the number of start points.
Tbd.	Not sure if the default start points are appropriate for later-era starts.

SPaH suppresses the difficulty adjustment of AI start points.

Rationale	When playing with SPaH, any free initial items and any AI head start in terms of initial items should be covered by the start points configured for the AI.
See also	[250c](#) disables free AI tech from difficulty setting when using (any) Advanced Start option.

(Just restating what's already described in the SPaH [chapter](#)) SPaH gives the human civs normal starts with e.g. a Settler and Warrior/ Scout in the Ancient era, and Advanced Starts to the AI. If less than 10000 start points are entered, all AI civs get the same number of points (flat distribution). Otherwise, if a 5- or 6-digit number is entered, the AI civs receive unequal amounts of points. The first 4 digits determine the max number of points. This is assigned to the AI civ in the bottommost slot. The last two digits are interpreted as a percentage of the max. The resulting min number is for the AI civ in the topmost slot. The others receive values in between min and max, linearly distributed and slightly randomized. The AI start points can be reviewed on the Settings tab of the Victory Screen.	There is no SPaH option; all civs, human and AI, receive the same number of start points.

Tbd.	The unequal distribution is linear between max and min, but also based on pairs that tend to have similar point values. Originally, this was supposed to make the assignment of start plots easier. The human civ and the bottommost AI civ were placed first, and the other civs were evenly distributed in between. This doesn't generalize well for multiplayer, so I've abandoned it. Given the current start-plot assignment algorithm, it would be conceptually simpler to use a linear distribution without pairs.

Config	The randomization of start points is configurable in `GlobalDefines_advc.xml`.

SPaH is factored into the game-end score. More specifically, the difficulty factor, normally between 0 and 90 (see 250a), is increased based on the start point distribution. For example, a Noble game has normally a difficulty of 30; with 1000 start points for every AI civ, that difficulty is increased by 20, which results in the same score as a Monarch game without SPaH.	Game-end score doesn't account for Advanced Start.
Rationale	The end-score function has numerous issues, but I think players still care about it a little bit as a rough indicator of how well they did. So, the formula doesn't have to be perfect, but, if start points work as a handicap, they should be accounted for somehow. I haven't thought about scoring in multiplayer games – not important enough to worry about.

AdvCiv	*BtS*
Default number of start points for Ancient-era start set to 400.	600
Rationale	I want this default for SPaH. For regular Advanced Start, 600-800 would be better (considering that 250c increases start point costs), but this can't be set dynamically.

250c	Changes to (regular) Advanced Start
No impact of difficulty setting on the human start points.	Extra start points for human civs on low difficulty settings (150% on Settler), reduced start points on high difficulty (75% on Deity).
AI start point bonuses increased to Prince 125% Monarch 145% Emperor 175% Immortal 210% Deity 270% (and they're actually applied). AI start point penalty added on difficulty levels below Noble.	Set to 110% 120% 135% 150% 170% but aren't applied because of a bug.
Rationale	Want to reduce the number of modifiers that affect start points. Instead of giving human civs fewer points on higher difficulty settings and AI civs more, I'm giving the AI civs a lot more. This is also to account for the free AI tech that is no longer granted in Advanced Start (see below). The higher amount of start points for human civs on Prince and above is countered by an increased point cost for units, buildings and cities (see below).
Credits	I had at first forgotten (not bothered?) to add an AI penalty on the low levels; added them after seeing this Kek-Mod commit.
No free initial items in Advanced Start except for the technologies of previous eras when starting in the Classical era or later and civ-specific tech when starting in the Ancient era.	No free units, but free technologies from the difficulty setting are granted in addition to start points.
Rationale	Shouldn't treat free (AI) tech differently than free units. Obviously, free tech from earlier eras needs to be granted, so my solution isn't entirely consistent either.

Charge 1.5 start points per production (units, buildings, cities).	The ratio is 1 for 1. 1 Food (population) costs 1.5 start points though.
1.5 start points per 1 culture.	1 culture costs 2.5 start points.
Revealing a tile costs 1 start point plus 1 per every 25 revealed tiles. Advanced Start normally reveals at least 50 tiles upfront. Although AdvCiv doesn't count these fully, the cost normally starts at 2.	$2 * (100 + 3 *$ number of revealed tiles$) / 100$ rounded down, i.e. 5 initially.
Can't reveal Ocean tiles that aren't adjacent to Coast when starting in Renaissance or earlier.	Can reveal any tile that isn't adjacent to a previously revealed tile (no change).
Start point cost for routes and improvements reduced to 67%.	E.g. a Mine costs 24 start points, a Farm 30.
Techs cost 1 start point per research point (no change).	

Rationale	Units seemed extremely cheap compared with tech. It's not that dangerous because units can only be placed in cities, and each city has a limit of two military units; still, makes lots of Workers and Settlers a too obvious choice. 1 start point essentially corresponds to 1 gold, so 1.5 per 1 production is still a bargain. Should perhaps be 2 for 1 – not sure if there's a good reason to make techs relatively unattractive during Advanced Start. Similarly, 1 revealed tile per 2 start points is still pretty expensive; seems better to buy Settlers and found later than to pay some 50 points extra for revealing the surroundings of a potential city site.
	A Mine for 24 is too expensive when a Worker costs 60. Now Workers are at 90 and Mines at 16.
	With cheaper visibility, it could make sense to reveal a diagonal across the entire map under the BtS rules. That would spoil all the surprises, so I'm prohibiting the oceans from being revealed.
	2 culture per 3 start points is still expensive – except for the first border pop, which I don't want to turn into a complete no-brainer.
Config	Costs for culture, revealed tiles and Worker builds are set in `GlobalDefines_advc.xml`.

Start points shown on the Custom Game screen are only auto-adjusted to match the start era. All start point costs are reduced based on game speed.	Shown points are adjusted to game era, game speed (silently) and world size. The impact of world size is minor; the multiplier is between 80% and 120%.

Rationale	A large world isn't necessarily less crowded than a small one. The BtS auto-adjustment is generally problematic because it overwrites any start point value that the player may have already entered. Also gets in the way of 250b – don't want the player handicap to increase with the world size.
	The silent game speed adjustment can be a bit confusing – the player ends up with a start point budget that differs from the value shown on the Custom Game screen. I think it's a bit more elegant to change the costs instead. Makes no functional difference (apart from rounding).

Dawn-of-Man Screen always shown. The "since time immemorial" text and initial techs are only shown when starting in the Ancient era. While in Advanced Start, scoreboard help text shows the player's leader.	Dawn-of-Man only shown for normal start, i.e. non-Advanced in the Ancient era. While in Advanced Start, players can only infer their leader from the text color or by using keyboard shortcuts to access Advisor Screens or Civ Description (Alt+D).
Rationale	Players need to know their civ and leader upfront, especially if it was set to Random on the Custom Game screen. The initial techs are meaningless when starting in a later era.
The start turn is based on the number of start points, more specifically, the maximum and the mean of the start point distribution. Only applies when starting in the Ancient era (and only when the average of mean and maximum is greater than 500). Example: If everyone receives 750 points (max=mean=750) on Normal speed, Emperor difficulty, then the start turn is 20, i.e. 1880 B.C.	The start turn only depends on the start era.
Rationale	Should match the historical time line better, and, in the case of SPaH, gives the impression that some of the AI civs have been on the map since 4000 BC, whereas the human civs are late arrivals.
Config	`INCREASE_START_TURN` in `GlobalDefines_advc`
No free food when population is added to a city through Advanced Start.	Whenever the population of a city is increased in Advanced Start, half of the city's food store gets filled. 0 food in newly placed cities.
Rationale	Perhaps the free food was supposed to make Granary useful in Advanced Start? But then the cost for increasing population should also be based on the present food store (and it isn't). Anyway, the rule is too obscure; no one expects free food when adding population. If adding population isn't powerful enough, one could always adjust `ADVANCED_START_POPULATION_COST`, but 1.5 points per 1 food seems fine now that 1 production also costs 1.5 points.
Credits	Kek-Mod (Git commit 1 2)
The Imperialistic trait reduces the point cost for placing cities except for the first city.	The Imperialistic trait reduces the point cost for Settlers but not for cities. Civs are forced to place at least one city before they can buy Settlers. When not in Advanced Start, the Imperialistic trait only increases the production invested in Settlers, not food. Thus, the production bonus is closer to 25% in the early game than to 50%.
Rationale	Inconsistent to apply the bonus to Settlers and not to cities. No bonus for the first city because, in non-Advanced Start games, Imperialistic leaders don't get an advantage from the initial Settler either. I'm OK with the 50% bonus (rather than e.g. 25%) because Imperialistic is a fairly weak trait.
Credits	Kek-Mod (Git commit); I've only added the exception for the first city.

The AI doesn't spend start points on culture in its capital unless it has a lot of points to spend.	The AI expands the borders of every city that it places.
Rationale	The border pop is normally a good investment but not really needed in the capital (although it's so cheap that this was barely worth changing).
Credits	Afforess (RoM: A New Dawn mod); SourceForge revision: link
Tbd.	Afforess makes some other changes in the Advanced Start AI code. I think these are mostly for dealing with very high numbers of start points, say, more than 10 000. A lot of magic numbers in his code that would probably have to be adjusted to AdvCiv. Either way, one thing that needs to be fixed: The AI currently places improvements (in `CvPlayerAI::AI_advancedStartPlaceCity`) before purchasing any techs. Should instead only reserve some points for improvements at first (could use the current code for that; just don't actually buy the improvements) and select and place them after purchasing tech.
Add "can't be removed" warning to building buttons and to tech tree button in Advanced Start.	Add visibility, add city and add culture have such warnings. Units, improvements, routes and population don't – and can be removed throughright click.
Rationale	Not obvious that buildings and tech can't be removed. (Could probably allow some of them to be removed, but I'm not going to bother with that.)
Tbd.	Should mention somewhere in help text that removal works through right click. Loading screen hint?
Units that the AI purchases for exploration during an Advanced Start receive the `UnitAIType` "Explore".	Units purchased for exploration receive the default UnitAIType. For Warriors, that's "Attack". Consequently, the Warriors aren't used for exploration but, at best, for patrols, and uusually they just guard the capital.
Rationale	It's embarassing when the AI fails to clear goody huts close to its capital.

250d	AI unit supply and upgrade cost not affected by game era
The AI advantages that are the same for all difficulty settings, namely the discounts on unit supply and upgrade cost, are unaffected by the AI game progress modifier (per-era modifier in BtS, per game turn since change 251).	The per-era modifier increases most of the AI advantages with each passing era.
Rationale	I wanted to apply the AI supply cost modifier to the gold paid for Pacifism (change 912b). The era progression complicates this. And perhaps AI upgrades also happen a bit too swiftly in the late game on Emperor and above. It's also conceptually simpler to have these two AI modifiers apply regardless of difficulty, i.e. also regardless of the per-era modifier (which does depend on difficulty). Look at them as an acknowledgment that the AI needs to have far more units than a human player in order to guard its cities. Note that AI upgrade costs still decrease over the course of a game (on Prince difficulty or higher) because the upgrade cost is computed based on unit production cost, and AI discounts on unit production increase as the game progresses.
See also	251 exempts all AI gold costs from game progress adjustments.
250e	Fewer AI freebies, especially units.

On Emperor difficulty and Immortal, the AI receives 1 free Scout.	2 Scouts; same as on Deity.
Rationale	To make Scouts more attractive for human civs; especially the initial Scout from Hunting. I don't think extra AI exploration units are a good way to ramp up the difficulty. Also note that, on Monarch, most AI civs don't start with Hunting and thus get 0 free Scouts. 0 to 2 is quite a jump when going from Monarch to Emperor. As for the extra Scout on Deity, it's worth noting that the AI won't use its free Archers for exploration.
On Monarch difficulty, AI civs that start with Archery but without Hunting receive an Archer as defensive unit and a Warrior as exploration unit. Those that start with Hunting receive a Scout instead of the Warrior.	Monarch AI civs without Hunting receive two Archers, one for defense, one for exploration. Those with Hunting receive an Archer and a Scout.
Rationale	A Scout is arguably less valuable than a second Archer, and starting with Hunting shouldn't be a disadvantage.
On Immortal difficulty, the AI starts with 2 free Archers (same as on Emperor) and on Deity with 3 free Archers.	3 on Immortal, 4 on Deity.
On Immortal, Barbarian cities start with 3 defenders.	4, same as on Deity.
Rationale	That many initial defenders just seem pointless to me.
See also	The table in the chapter about the SPaH option lists the AI freebies for each difficulty setting.
No free Agriculture for the AI on any difficulty level.	Free Agriculture on Immortal (in addition to Hunting and Archery) and on Deity (in addition to Hunting, Archery and The Wheel).
Rationale	It seems that the jump in difficulty from Emperor to Immortal is a bit too big in general; especially the first one and a half eras are tense, the player is easily boxed in. With the No Slavery option (912d) enabled, it seems very difficult to persevere on a continent shared with a single AI civ. Taking away a free tech seems like the best way to slow AI expansion down a little. Agriculture has the potential of speeding up a second AI settler considerably; I think The Wheel (on Deity) is a less explosive freebie, rather steering the AI toward early Pottery and Cottages, i.e. toward playing a long game.
See also	I guess changes to map sizes (137, 165), starting positions (027) and AI behavior have made it considerably more difficult to establish at least four decent cities in the early game. Converting AI worker speed increases into a human worker speed decrease (251) may also have had the (unintended) effect of hampering the early development of human civs more than that of AI civs. CFC post of mine (the long paragraph in the middle) about the state of Immortal and Deity in AdvCiv 0.99 (i.e. before the Agriculture change, but after the Deity Settler change), linking to two other relevant threads.

No second free Settler unit for the AI civs on Deity. I.e. the AI starting units are the same as on Immortal except for an extra Archer.	The second Settler means that the AI head start on Deity is much greater than the (already substantial) head start on Immortal. For a challenge in between Immortal and Deity, BtS players sometimes remove the second Settler; that custom difficulty level is sometimes referred to as "Demigod" (that had also been the name of an official difficulty level in Civ 3).
See also	The Deity research rate set by 251 is adjusted to this change.
Rationale	With the second Settler unit, Deity wasn't really playable on a normal map with the default player count as the human player was immediately boxed in. I'm not sure if Deity is now a reasonable difficulty to play on in AdvCiv. (Ultimately, i.e. after some more balance changes, I hope that even the most competent players would be able to find a challenge on Immortal, if not Emperor. Deity would then just serve as a demonstration of what happens if the AI advantages are increased beyond Immortal.)
Tbd.	Insert a difficulty "God Emperor" in between Emperor and Immortal, and boot Settler? The total number of levels should stay the same for replay compatibility (see 106i). I've created a Git branch for God Emperor.

251	Pacing adjustments for difficulty levels
See also	910 adjusts the per-era tech modifiers (which apply to everyone, not just the AI) and the game year increments.

AdvCiv		*BtS*
The start turn is advanced by 10 (on Normal speed) for every free Worker and Settler (beyond the first) that the AI receives. This only applies when playing without Advanced Start (about the start turn in Advanced Start, see 250c). The "Settings" tab (Victory screen) shows the start turn number when it isn't 0 (for any reason).		The start turn is only affected by start era, not by difficulty.
Config	`INCREASE_START_TURN` in `GlobalDefines_advc.xml`	
AI tech costs are reduced on the lowest four difficulty settings, and increased on the highest three. I've adjusted the human tech cost modifiers for these seven difficulty settings so that the ratio between human and AI tech costs remains approximately as in BtS.		AI tech costs are not affected by difficulty. Human tech costs are reduced to 60% (Settler) to 90% (Warlord) on difficulties below Noble, and increased to 110% (Prince) to 130% (Deity) on difficulties above Noble.
Rationale	The overall progress had been too fast on difficulties above Monarch and too slow on difficulties below Prince. It's an aesthetic issue (AI tech matching game date), but also a matter of pace: How much time there is for deploying combat units until another tech is discovered. I don't think that, say, Warlord and Immortal should have fundamentally different pace (that's what the game speed setting is for). Deity still gets ahead of the historical time line, but not all too far.	
See also	karadoc seems to have thought so too; in K-Mod 1.45, he tied inflation to the global technological progress. I don't like this solution; see the discussion here.	

	Civ 4 Reimagined appears to do something similar to my change ("tech costs for all players scale with handicap setting;" Bitbucket link). And Caveman to Cosmos also (SourceForge link; 12th bullet from the top).
Config	`AIResearchPercent` and `ResearchPercent` in `Civ4HandicapInfos.xml`

Starting on Emperor difficulty, production costs, city growth thresholds and Great Person (GP) thresholds of human civs are increased by 10% (Emperor), 20% (Immortal) or 30% (Deity). Exception: Building production costs aren't increased as much. AI production discounts are lower than in BtS, e.g. just 5% on Emperor (at game start).	Human production costs and growth thresholds aren't affected by difficulty, and neither human nor AI thresholds for GP are affected by difficulty. The AI production discounts are e.g. 15% on Emperor and 40% on Deity. These discounts increase over the course of the game, so that, by the Modern era, AI production costs on Emperor are only about 70% of the normal cost.
The AI discounts for wonders are the the same as for other buildings and units. (Except for a 5% production discount on Prince difficulty that doesn't apply to wonders.)	At game start, the AI receives no production discounts on wonders. The per-era discounts do apply to wonders.
The modifier for human city growth only affects the base threshold (20 food) for growing a city. That threshold increases by 2 food per population regardless of the difficulty setting. The AI growth modifier works as in BtS, i.e. it also applies to the extra food from population.	The AI city growth threshold is decreased by 5% on Monarch, 10% on Emperor, 15% on Immortal and 20% on Deity; same in AdvCiv.
The production (and Advanced Start) cost of Settlers is partly affected by the growth modifier (as in BtS) and half affected by the production modifier. This results e.g. in a cost of 130 on Deity.	For each 1 food that the initial city growth threshold differs from 20, the cost for a Settler is adjusted by 5%. Production modifiers don't apply.
The GP thresholds of the AI are adjusted by the same percentages as the city growth thresholds.	
On the lowest two difficulty levels, human production costs and growth and GP thresholds are lowered. Free happiness and health on Settler difficulty lowered by 1; now the same as on Chieftain.	

Rationale	To reduce overcrowding of the map on the high difficulty levels and to match the increased research costs. If only research becomes more expensive as the difficulty level increases, the game balance skews toward warfare in a similar way as on Marathon speed. Perhaps Noble difficulty (on Normal speed) makes it a bit too rewarding to focus on research, but the human research cost modifier on Deity (now 160%) would definitely be too high without a production cost penalty, and on Emperor (125%) arguably also.
	Similarly, the human costs for growing cities and GP need to match research costs to an extent. The game speed settings, for comparison, have growth and GP modifiers too, and also modify Worker speed (see below), Cottage growth, culture level thresholds, chopping production, hurry production and GP effects. I don't see a need for making all these values dependent on difficulty; it's actually fine if chopping and Slavery become less efficient on the high difficulty levels.
	Building production costs don't get adjusted as much as unit (and project) production because the full adjustment would make too many buildings categorically unusable on Immortal and Deity. Buildings tend to provide predictable but slow returns, which isn't

	good enough on the highest difficulty levels. Also, keeping relatively high AI discounts on (non-wonder) buildings isn't much of a problem; won't contribute much to Stack of Doom problems.
	No production/ growth/ GP penalties on the medium difficulty levels because a small (say 5%) increase would be barely noticeable due to rounding (see next blue box).
	Given the lower AI production discounts, exempting wonders should no longer be necessary. I'm only doing it on Prince to differentiate Prince from Monarch; these two levels would otherwise have the same AI production discounts (5%).
	The human growth penalty is applied only to the base food needed for growth because I want to keep the formula for the food per population simple (2 food per population, same as the food consumption). For the AI, simplicity isn't so relevant, and a growth bonus that applies also to the food per population helps the AI in the midgame and late game.
	The GP discounts and penalties make the high difficulty levels harder and more unfair than in BtS. Changes to the per-era modifier (see below) might even this out.
Config	I've added tags `TrainPercent` (for units), `ConstructPercent` (for buildings, incl. wonders) and `CreatePercent` (for projects, incl. world projects) to `Civ4HandicapInfos.xml`. These apply to all civs that play at the respective handicap (AI civs play at Noble unless a scenario gives them a different handicap).
	For GP and growth thresholds, I've added `GPThresholdPercent` and `BaseGrowthThresholdPercent` (for all civs) and `AIGPThresholdPercent` (for AI civs).
See also	Since these modifier changes don't affect Barbarians, change 300 decreases the Barbarian activity on the high levels a bit.
	101 gives AI-owned cities a somewhat smaller revolt chance. This stacks the game a bit further in favor of the AI (but I don't think it's a significant factor overall).
Tbd.	I'm not sure if the Processes (e.g. Wealth) need to be difficulty-adjusted. When both research costs and production costs increase with the difficulty level, one would think that the conversion rate between them can stay the same. But I think there's still a problem of buildings becoming less attractive as the difficulty level increases, and the processes are generally recognized as more efficient than buildings, so perhaps something like 100% Prince, 90% Monarch, 80% Emperor, 75% Immortal, 70% Deity, maybe coupled with a 20%(?) boost at Computers. Perhaps not good to punish players for running out of things to produce in the endgame.

City maintenance and unit cost modifiers increased a bit on difficulty settings below Emperor. Free wins against Barbarians reduced by 1, meaning that Noble is now the highest difficulty level with a free win (was Prince in BtS). Slightly reduced tech costs on Noble difficulty to keep the pace of the game the same.

Rationale/ See also	708: Aimed mainly at experienced players that combine a medium difficulty level with the R&F game option or some other special-challenge game option.
	Since AI civs plays on (player handicap) Noble difficulty, this change hurts them a little and thus makes it a little easier to compete with them on (game handicap) Emperor and above. Shouldn't make a big difference though – due to the AI discounts on Emperor and above, AI expenses should still be pretty painless. Hopefully, a single free win on Noble is still enough to keep the AI out of trouble in the very early game.

Unlike the other human production penalties, those for team projects only apply on Immortal and Deity difficulty. On Emperor and below, the	No modifiers for human project costs. The AI modifiers for projects are equal to those for other types of city production.

AI receives higher team project discounts instead (the same as in BtS or similar).	
Rationale	Don't want to make Space victory even more grueling than in BtS. See `CultureLevelPercent` below about balancing Space and Culture victory.
Human worker build times are increased by 10% on Emperor, 15% on Immortal and 20% on Deity. The resulting build times are rounded down to to a multiple of 50 before modifiers from game speed and start era are applied. The AI work rate bonus is 5% on Prince and increases in steps of 5 percentage points to 25% on Deity.	Human workers aren't affected by the difficulty setting. The AI work rate bonus is 10% on Prince, 20 on Monarch, 50 on Emperor, 75 on Immortal, 100 on Deity. Build times aren't affected by difficulty. Since the base build times (e.g. 200 time units for Road) are a multiple of the work rate of the Worker unit (100), even a 1% build time penalty results in at least one extra Worker turn for all builds.
Rationale	In the very early game, build times should somewhat match research times, especially for human civs. Otherwise, civs that don't receive any free worker tech can become unplayable and players can't afford to research any other techs. Also, the very fast AI work rates in BtS can help the AI get ahead too fast and could provide an extra incentive for attacking AI Workers.
	Later on, quick AI build times mostly mean that the AI quickly improves and routes every tile, which looks ugly and rather unintelligent.
	Given the way that the number of build turns is calculated, a work rate penalty would result in an extra turn for everything; something smoother is needed so that e.g Road can still be built in 2 turns, but Farm may require an extra turn. The build time increase with rounding, albeit complicated, accomplishes that. Work rate *bonuses* are fine in this regard.
Config	New XML tag `BuildTimePercent` in `Civ4HandicapInfos.xml`
Human production and research costs and GP thresholds are rounded to the nearest multiple of 5.	Human production costs and GP thresholds are only modified by game speed and these modifiers usually result in multiples of 5, though e.g. an Archer costs 37 production on Epic speed. The research cost modifier from difficulty can result in odd research costs.
Rationale	Easier to remember and calculate. The game balance isn't so delicate that +/-2 production will make a big difference.
The AI discounts on production costs, research costs and city growth thresholds increase by 1 percentage point every 100 turns on Prince difficulty, every 50 turns on Monarch, every 33 turns on Emperor, every 25 turns on Immortal and every 20 turns on Deity. If the game doesn't start on turn 0, these increments are applied retroactively, i.e. the current game turn is decisive, not the number of elapsed game turns. Also, the numbers are adjusted to the game speed setting (to `VictoryDelayPercent` to be specific). These increments based on the game turn replace the BtS per-era modifiers.	Whenever an AI civ enters a new era, its discounts on production costs, city growth, expenses (inflation, civic upkeep, unit cost, unit supply), unit upgrades and war weariness are increased by (multiplicative) 1% on Prince, 2% on Monarch, 3% on Emperor, 4% on Immortal and 5% on Deity.
See also	250d: AI unit supply and upgrade costs are independent of the game progress.
Rationale	The per-era mechanism is self-reinforcing because the sooner an AI civ reaches a

new era, the sooner it benefits from increased discounts; such snowball effects are undesirable. Using the game era instead of civ era would also be problematic because of possible human manipulation, and any average gets distorted by vassals/ small civs. The game-turn-based mechanism doesn't have these problems and works more smoothly. One potential issue is that high-difficulty games tend to reach the endgame eras earlier than low-difficulty games, meaning that the game progress adjustment may not take full effect. However, the current difficulty-based research cost increases seem to be quite effective at giving games across all difficulty levels a similar length.

Now that AI research costs are affected by the difficulty setting, it's conceptually simpler to adjust the AI research cost to the game progress than to adjust the various expenses. Also, lower expenses have side-effects on the civics that the AI chooses and how many units it trains. As for war weariness, I don't think that's an issue for the AI even in the late game. As for GP thresholds, I worry that progressively lower thresholds would draw too much attention to the AI discounts (as every GP birth is announced).

Additive modifiers seem a bit simpler to me than multiplicative ones and lead to fewer rounding artifacts.

I'm not sure if these changes lead to higher or lower AI bonuses overall. For example, if an Immortal game reaches the Modern era by turn 350, the AI growth (threshold) modifier is now (85-350/25)%=71%; in BtS it's 85%*(100-5*4)%=68%, but also 71% just before entering the Modern era. It seems that the BtS effect is generally a bit greater. This should be evened out by the new GP threshold discounts/ penalties.

Config	I've renamed `AIPerEraModifier` in `Civ4HandicapInfos.xml` to `AIHandicapIncrementTurns` and changed its semantics, so it's not possible to return to the BtS mechanism through XML.

(Human) civic upkeep increased on all difficulty levels below Monarch; now the same progression as inflation.	The civic upkeep modifier is 5 to 10 percentage points smaller than the inflation modifier. On Monarch and all levels above, the inflation modifier is 100%.
Rationale	Civic upkeep is generally too insignificant.
See also	570 deals with other changes to expenses.
Config	`CivicUpkeepPercent` in `Civ4HandicapInfos.xml`

The AI civic upkeep and unit cost modifiers on Deity are 75%.	60%
Rationale	Don't want the Deity AI to favor high-upkeep civics and don't want to encourage it to train very large armies.
Config	`AIUnitCostPercent` and `AICivicUpkeepPercent` in `Civ4HandicapInfos.xml`

On Monarch and above, culture level thresholds (except "Fledgling") are adjusted to the game's difficulty level.	Culture level thresholds are only adjusted to game speed and the "No Espionage" option (disabled by change 309).
Rationale	If tech costs are increased, then the Legendary culture threshold will also have to be increased (though I guess not as much?) to keep Space and Culture victory balanced. No special adjustment for the AI; I reckon that the other AI advantages give the AI a sufficient edge when it comes to Culture victory, and explicitly making a victory condition easier for the AI would be (perceived as) a new quality of AI "cheating."
Tbd.	Not sure if the base value of 50000 (set in `Civ4CultureLevelInfos.xml`) is well-balanced. The tech cost changes under 910 have made Space victories costlier, but

	K-Mod has nerfed various important culture sources; so perhaps it's OK. In any case, I'd like to keep it at 50000 and instead nerf or boost the available sources of culture.
Config	`CultureLevelPercent` in `Civ4HandicapInfos.xml`
See also	[126](#) adjusts the culture level thresholds to the game's start era.

AdvCiv	*K-Mod*
Great Work ("culture bomb") culture increases by 700 per era (of the owner of the Great Artist), beginning with the Ancient era. And hover text for the Great Work button shows the culture per era.	Starts at 0 culture in the Ancient era, then increases by 800 per era. (4000 flat in BtS, i.e. regardless of the era.) K-Mod has changed the ability text in Civilopedia, but the action button only shows the total culture based on the current era.

Rationale	The K-Mod change is apparently aimed at border spread, and it's probably true that 4000 culture in the Ancient era is – perhaps not overpowered compared with, say, an Academy – but not well-balanced and not enjoyable to play against. However, when focusing on a Culture victory by the midgame, the reduced culture is quite an impediment. While, in my opinion (from a game designer's point of view), teching to the Modern era should be the standard approach when playing at a challenging difficulty level, teching only to the Industrial era should also be worth considering. I think the K-Mod progression hurts that approach a bit too much. (Making settled Great Artists a more worthwhile alternative would be noble goal, but I don't think it's close even with the K-Mod progression.) Starting at 0 culture in the Ancient era also seems wrong to me for mod-mods (in BtS/ AdvCiv, an Ancient Great Artist is impossible to obtain).
Config	`iGreatWorkCulture` of `UNIT_ARTIST` in `Civ4UnitInfos.xml`.
Tbd.	Shouldn't be tied directly to the owner's era but to something smoother, like the other Great Person effects. Because, so far (and in BtS), players don't need to plan for era transitions and don't need to be aware which techs will trigger them. I'd rather keep it this way than clutter the UI with easily visible info about tech eras. Should announce clture bombs in revealed cities (also in observer mode): "The Great Artist Homer has dedicated his masterpiece to the city of Beshbalik." With a Great Work button icon, perhaps a sound too (or play that only for the city owner).

AdvCiv	*BtS*
Decreased the base threshold (Prince difficulty and below) for Legendary culture a bit – except on Quick speed, where the threshold is now about 2/3 of the Normal-speed threshold. Also increased the thresholds for all other culture levels on Quick speed.	The threshold is 50k on Normal speed, three times as much on Marathon, 1.5 times on Epic and 0.5 times on Quick speed.

See also	[Discussion](#) with crullerdonut about Culture victory (starting after the 5th quote box) [910](#) increases late-game tech costs and reduces research rates
Rationale	Given the current tech costs and research modifiers in AdvCiv and the +1 research to specialists at Scientific Method, the rate of tech discoveries in the late-game should be about 75% of the BtS rate. The typical culture modifier when going for a Culture victory in BtS is 100 to 150 from 2 to 3 Cathedrals, 100 from Free Speech, 50 from Broadcast Tower; that's 250 to 300%. In AdvCiv, that modifier is 80 to 120**(*)** plus 50 plus 50 = 180 to 220%, i.e. ca. 72% of the BtS rate. This sounds almost well balanced assuming that research is the bottleneck for a Space victory – and assuming that Culture vs. Space is fairly well balanced in BtS, i.e.

that researching all the techs that a Culture victory doesn't require tends to take a player focused on a Space victory about as long as it takes a Culture-focused player to accumulate 50k culture after cranking up the culture slider.

The main factor that does seem to make Culture victory more difficult overall in AdvCiv than in BtS is that UWAI makes it more difficult to avert late-game wars through diplomacy; that may make it unviable to deprioritize research at the end of Renaissance. Civilized Jewelers at Corporation can't make up for that (not sure if it's worthwhile for a Culture victory at all).

I would prefer to balance this out by slapping +25% culture on some secular building (making Culture victory a bit less dependent on religions), like K-Mod did with Colosseum – but Colosseum really shouldn't be a source of culture; don't want to blur the distinction between amphitheaters and theaters. Another candidate is Market, but that's also strenuous, and, since capitals usually construct a Market before long, it would further increase the culture pressure from capitals, which already seems rather too great. Theater would be the most intuitive place, but then the building should either not grant flat culture or be a good deal more expensive. Either change would make Theater far less attractive in recently conquered cities.

For lack of a better alternative, I'm adjusting the Legendary threshold. The difficulty adjustment already means that 50k is no longer a number that players can rely on, so I don't feel too bad about this.

(*) In AdvCiv 1.0, I've restored the Cathedrals to 50%. The calculation above doesn't take into account that religious culture from Sistine Chapel no longer doubles and that Great Artist culture is only 3500 (after an adjustment on my part, would've been 3200 in K-Mod) unless the player techs into the Modern era. Moreover, if the player stops teching at the onset of the Industrial era (which I'd like to remain a consideration), the +50% culture from Broadcast Tower also can't be counted.

Quick speed: All the speed-adjustments relevant for producing city culture seem to use a multiplier of 67%, and the CFC posts that I was able to find on that topic paint Quick-speed Culture victory as a rather cheap way of winning. Seems that Firaxis went with 50% simply because that results in more aesthetically pleasing (and perhaps more easily remembered) numbers.

297	*(unassigned)*

298	Tweaks to opportunistic AI attacks (`AI_anyAttack`)
AdvCiv	*BtS*
Non-lethal units do not go out of their way to attack weaker enemy units – unless there are lethal units nearby that might finish the job.	Sometimes, individual siege units move into enemy lines in order to make some attack with high survival odds.
Rationale	Even when a non-lethal attack is safe, the attacker is likely to be vulnerable to counterattacks; not worth it.
Credits	CFC user eXalted reported the problem and provided a savegame: here

299	AI tweaks for healing
When deciding whether to heal a unit, the AI takes into account the defensive modifiers of the current tile. (Does not apply to automated human units.)	
Credits	Inspired by similar changes in "More Naval AI"; Git commits 1 2
See also	300: AI prefers to guard city sites from tiles with a defensive modifier

300 et sq.	Overhaul of Barbarians; see chapter Revised Barbarians.
Config	A few parameters in `GlobalDefines_advc.xml`
Tbd.	Should move all the Barbarian placement code (cities, animals, non-animals) to a separate class. It's not particularly messy as it is, CvGame just does way too much. It's a lot of tedious work however.
See also	250e reduces the number of defensive units placed in Barbarian cities on Immortal difficulty.
300	The fundamentals: gradually increasing activity, ships placed per shelf, food yields matter, fogbusting nerf, game speed scaling, city placement bugfix, New World Barbarians, no cooperation between Barbarian cities, misc. AI improvements vs. Barbarians

AdvCiv	*BtS* (K-Mod hardly changes Barbarians)
Barbarian activity reaches its peak when 56% a continent are owned by civs (or Barbarian cities). With Raging Barbarians, the activity peak is a little later.	Barbarian activity peaks shortly after they first appear, typically around turn 50.
The difficulty setting determines on which turn Barbarians first appear. I've increased the start turn set there by 5 for all levels, but, on high difficulty settings, they still appear quite early, e.g. after 15 turns on Deity. They initially appear only in small numbers though, and don't immediately attack cities.	Barbarians can't appear until there are 1.5 cities per civ. Thus, the early start turns set for high difficulty settings through XML have practically no effect.

Rationale	Since Barbarians start slowly, the 1.5-cities rules isn't needed anymore.
	I had used 67% until AdvCiv 0.98. That may have matched the time line for the "age of migration" better, but, considering the higher default player count (137), often resulted in no sense of urgency while founding a 2nd and 3rd city and made it undesirable to produce Warriors while growing the capital to size 3. (It's often better to grow to size 3 even if another Warrior isn't needed anytime soon, but it feels bad to produce unnecessary units.)

Barbarian ships are created for each continental shelf, i.e. the ring of coastal tiles surrounding each land mass (or enclosed in the case of inland seas).	Barbarian ships are created for each water body; usually, there is only one large water body.

Rationale	To prevent neglected shores on one continent from affecting Barbarian activity on the shores of another continent, and to prevent Barbarian ships from piling up near remote islands (as they sometimes do in BtS).

Tbd.	When a shelf is interrupted by Ice, Barbarian ships can stile accumulate in between Ice tiles. Need to treat such pockets as separate shelves (akin to what 030 does). As a temporary solution, no Barbarian ships are placed if the total size of a shelf is far greater than the number of tiles where a Barbarian ship can legally appear.
(Disabled) When playing with Raging Barbarians (RB), the Barbarian start turn is shown on the Settings tab of the Victory Screen.	The game never tells the player on which turn Barbarians start appearing.
Rationale	When starting in a later era or using SPaH, the game doesn't start on turn 0, and it's difficult to tell how many turns have passed. Update (v0.94): Not crucial to know because even Raging Barbarians start appearing gradually. And after the turn on which Barbarians could theoretically first appear (based on the difficulty setting), it may take another 10 or 20 turns until a unit actually enters visibility, so the turn number can be misleading.
See also	251 shows the start turn on the Settings tab if it isn't turn 0.

When computing the target number of Barbarian land units and cities per continent, half of the coastal water tiles in the surrounding shelf also count, and non-habitable tiles are disregarded. I refer to tiles as "habitable" if they have a positive natural food yield when ignoring hill yield changes.	All land tiles count equally, and water tiles count only for sea units.
Barbarian units can't appear on non-habitable tiles. On "non-arable" tiles, i.e. tiles with a yield sum of 1 or 1.5 (counting commerce as 0.5), Barbarians are far less likely to appear than on tiles with higher yields. I'm computing the yield sum twice, once including and once excluding features, and take the minimum. This way, all Tundra and Jungle without (revealed) resources is treated as non-arable.	Tile yields have no impact on Barbarian unit placement.
Civ units only prevent Barbarians from appearing on visible tiles.	Barbarians can't appear in a 5x5-tile square surrounding each civ unit ("fogbusting" range). Visibility practically doesn't matter.
Animals only appear on tiles with a positive food yield or freshwater.	A passable tile not visible to any civ is chosen at random, then an animal suitable for that tile's terrain is chosen.

Rationale	Want Barbarians to work as a corrective for starting positions that have much room for expansion. To this end, fogbusting needs to be nerfed, and decent tiles need to be distinguished from bad tiles.
	It's also rather implausible that large numbers of Barbarians would appear in the polar region or desert. The early civilizations were frequently raided by pastoralists from marginal (but not desertic) lands, which is why I'm not distinguishing between e.g. Plains, Grassland and Flood Plains.
	"Habitable" tiles: Until AdvCiv 0.97, those were tiles with a positive final food yield (including terrain, features, hills, improvements). That was nice and simple, but I think Plains Hill not spawning Barbarians is counterintuitive and treating Tundra Hill differently than flat Tundra is also dubious. "Arable" tiles had included river Tundra until AdvCiv 1.0; that seemed like a bit of a gotcha.
	Animals: A problem in Earth scenarios; the Sahara gets filled up with Lions. One could look at this as a representation of the harsh conditions there, making the desert difficult to explore, but, ultimately, the lion-filled Sahara looks too silly, and the Sahara lions prevent animals from spawning elsewhere.
See also	Change 304 also affects the per-tile placement probabilities, and the yield-based changes are implemented based on that.

Barbarian creation rate adjusted to game speed. The target number of Barbarians per continent (upper bound) remains independent of game speed. Barbarians start appearing roughly in the same year for all speed settings.	Creation rate and target number are independent of game speed. Barbarians start appearing later on slower settings.

Rationale	On Epic and Marathon, it takes civs longer to train units, but Barbarians are (re-)placed just as quickly as on Normal and Quick. The original developers may have intended to even this out through a later start year. This doesn't work well – BtS Barbarians are a lot harder to deal with on Marathon than on Normal speed, and I don't see why they should be.
	The creation rate does not take into account how densely or sparsely the civs are placed. It's plausible that more Barbarians appear when there is room for them, and I want to allow players to create games with extra tough Barbarians by placing civs far apart.

The creation rate of Barbarian cities increases slightly with the game era.	Static creation rate configured in XML.

No Barbarian units are created on continents without civ cities (nor in the surrounding waters), but Barbarian cities appear earlier on such continents and more densely than in BtS. If a continent (or shelf) becomes very densely populated with Barbarian units, the game starts culling them. The greater the density, the higher the per-turn probability of removing a unit. The Barbarian AI never disbands units for financial reasons, and Barbarians are immune to strikes.	Plenty of Barbarian units are created on continents without cities because these continents have 100% unowned tiles. While Barbarian units are created independently for each continent, cities are placed globally, and the game prefers to place them on continents with civ cities. The game might eventually remove Barbarian units through strike, and the Barbarian AI can disband units. Not sure how frequently that happens; perhaps not at all.

Rationale	Want terrae incognitae to look more like in Colonization, i.e. with lots of Barbarian cities but not so many units. The cities actually produce too many units, hence the scrapping mechanism; it's also a general safeguard against implausibly large Barbarian stacks. Could interpret the removed units as a result of Barbarian infighting. Disbanding based on finances isn't good because it may leave high concentrations in some areas, and because Barbarians aren't supposed to have an overarching economy.
Tbd.	Barbarians can currently be removed from tiles visible to a player, which is a bit jarring. Should show a message then, e.g. "A Barbarian Galley has been destroyed as a result of infighting." Can't simply exclude visible tiles from removal: In one game, for example, Barbarian Galleys kept spawning around an uninhabited continent and moved from there to an inhabited continent. In such a case, naval stacks of arbitrary size can form on visible tiles. Or simply program the Barbarian AI not to move into overcrowded non-city tiles.

Barbarian units are never upgraded.	Upgrades are possible using commerce from Barbarian cities but seem to happen rarely because resource requirements are checked when upgrading; cf.

Rationale	Don't want a Barbarian economy. Outdated units are still killed eventually, either in combat or removed by the scrapping mechanism above. A mixed bunch of Barbarian units is more flavorful than a uniformly upgraded army; I imagine that Barbarians acquire sophisticated weapons from the civs (through trade) but also continue making their traditional weapons. It can happen that civs keep trained and spawned Barbarians on a continent in check, thus preventing scrapping, but don't attack (all) the Barbarian cities. Then, early-game garrisons will remain unupgraded indefinitely, which starts looking weird once the Barbarians reach Gunpowder. Too rare to worry about.

Fixed two BtS bugs that lead to Barbarian cities being placed either in completely arbitrary locations (like a snow island without seafood), or in ideal locations (as far as the AI is able to figure those out).

See also	Thread on CFC

The aggressiveness of Barbarian units escalates independently for each continent based on the number of local civ cities.	There are essentially three levels of Barbarian aggressiveness: Initially, Barbarians avoid civ cities, then they attack them opportunistically, and eventually they seek cities out. This escalation is based on the *total* number of civ cities, and affects all continents alike.
Barbarians avoid approaching civ cities on continents with more Barbarian cities than civ cities, even when playing with Raging Barbarians (RB).	With RB, the Barbarians usually seek out cities right away. Without RB, only the number of civ cities is decisive, not the ratio of civ cities to Barbarian cities.
Rationale	To remove implausible repercussions of civ development on one continent for the Barbarian behavior on another continent. New World Barbarians shouldn't immediately attack colonies. Now only turn aggressive when half the continent is colonized.

Barbarians that look for a city to attack ignore the per-landmass target city of the Barbarian player unless there is at least one Barbarian city on the landmass.	Barbarians have a per-landmass target city like all other players. Units looking for a city to attack try to either attack that city directly or to capture nearby cities.

Rationale	I haven't looked into the details, but I expect that computing a target city for a player without local cities would have a pretty arbitrary result that would lead to arbitrary (and possibly unfair to the owner of the target city) Barbarian behavior.
	Not sure what else the target city is used for, so it seems safer not to just set it to `NULL` for landmasses without Barbarian cities.
	The above is obsolete. As of v0.99, Barbarians don't have a per-continent target city at all.
See also	The "More Naval AI" mod increases the impact of randomness on the choice of Barbarian target cities. ([Git commit](#))

For Barbarian units, the survival odds threshold for attacking from within a Barbarian city ("leave attack") is halved.

Rationale	When an AI civ brings an inadequate attack stack to a Barbarian city, there is often a lengthy standoff (since they can't make peace). This change may help a bit. That said, Barbarian Archers will often have near-0 odds, so I doubt that this change will make a big difference.

Barbarian Workers don't connect cities with roads, chop fewer Forests and prioritize production.	Barbarian Workers build road networks between Barbarian cities and builds lots of Cottages.

Rationale	A road network makes it look like a Barbarian empire, but it's supposed to look like independent cities. Cottages are nice for pillaging, but don't make much sense for the Barbarians. Chopping: Don't want players to discover a deforested New World.
See also	Kek-Mod has made a similar change (Git commit [1](#), [2](#), [3](#)); haven't merged it because I still want Barbarians to build *some* Cottages.
Tbd.	Barbarian Workers still tend to gather in a single city once all tiles are improved.
	Perhaps `CvPlot::isBonusNetwork` should not return `true` for unowned tiles if `eTeam==BARBARIAN_TEAM`. Currently, Barbarian cities can share strategic resources along unowned coast once they have Sailing. Kind of inteeresting. Perhaps leave it as it is.

If RB is enabled, AI civs assign fewer units to guard cities unless threatened militarily by another civ. New AI routine for guarding high-yield improvements against Barbarians.	The AI defends its cities against Raging Barbarians but doesn't defend crucial improvements.
	Often allows Barbarians to move onto Mines or onto improvements across a river, negating the 25% combat bonus that the AI gets against Barbarians on all difficulty levels.

Rationale	Also frees up units for guarding future city sites, which is similar enough to human fogbusting.
Tbd.	Doesn't help much. Early on, when it matters most, the AI still needs most of its defenders to protect its cities, and doesn't reliably identify the most precarious improvements. Would be better to move from the city onto a threatened improvement only when a Barbarian unit approaches. Need to be careful not to expose AI cities to Barbarian or human attacks though.

See also	[315](#) gives Scouts a defensive bonus against Barbarians and allows them to guard city sites.
The AI only guards proper city sites, i.e. tiles that it is presently willing to settle.	Willing to guard any tile with a positive found value. Found values are 0 near tiles that the AI currently wants to settle, so the guarded tiles are either proper city sites or sites in an area not currently worth settling (too remote or too marginal).
Rationale	Don't want the AI to guard remote city sites because this rarely helps the AI civ and may help other civs (by keeping away Barbarians).
AI may guard a tile adjacent to a proper city site if that tile has a higher defensive bonus or visibility.	Only city sites can be guarded.
Rationale	The main point of guarding city sites is fogbusting; should prefer hills and forest/jungle for that. Also avoids telegraphing to humans where exactly the AI is going to settle.
AdvCiv	*K-Mod*
AI Workers avoid tiles adjacent to a Barbarian border only if there is a concrete danger.	AI Workers avoid tiles at a hostile border even if those tiles aren't immediately threatened (as far as I understand the code).
Rationale	A single Barbarian city isn't very likely to disturb Workers or pillage improvements. And it can take the AI a long time to conquer a neighboring Barbarian city; don't want to leave tiles unimproved during that time.
AdvCiv	*BtS*
AI civs are more willing to target Barbarian cities on other landmasses when there is nothing else to do or when it doesn't take much effort.	AI civs hardly ever conquer Barbarian cities on landmasses where they don't have their capital.
Tbd.	The AI still attacks Barbarian cities too rarely, even on the same landmass. Just doesn't have enough city attackers at peacetime. May have to increase the probability of training city attackers based on number and attractiveness of Barbarian cities, perhaps through an AI strategy flag for fighting Barbarians/ Minor civs and/ or by setting the per-Area target city to a non-civ city (currently happens rarely, if ever). And the AI needs to be more reluctant to attack remote Barbarian cities. (Once this is fixed, the AI may no longer have to be extra reluctant to raze Barbarian cities; see [116](#).) For Barbarian cities on other landmasses, code for preparing and executing small-scale naval landings is needed. Warfare between civs on maps like Archipelago could also benefit from this.
AI civs use (at most) small stacks of a couple of units to "choke" (pillage, block tiles from being improved or worked) Barbarian cities that are too well defended for an attack	Normal choke behavior against Barbarian cities. Sizable stacks may camp indefinitely next to a Barbarian city on a hill defended by a few Archers. (Whereas, against an enemy civ, a peace treaty will often break the stalemate.)
Tbd.	Perhaps Barbarian cities shouldn't be choked at all; I don't think they rely on improved tiles much. The main benefit that I see is that the (quite aggressive) Barbarian tactical AI may launch premature sorties; so choking might help keep the Barbarian unit count in check. I'm also not sure if the choke routine is the only code responsible for standoffs at Barbarian cities.

AdvCiv	K-Mod/BtS
Barbarians raze cities based on the difference between the city owner's era (1 for Ancient, 2 Classical ...) and the city's population. The difference is multiplied with `RazeCityProb` =10%, the base probability for Barbarian razing. I.e., Barbarians only raze Ancient and Classical cities when they have to (size < 2), and can raze size-2 in the Medieval era and up to size-3 in Renaissance. I'm capping the era number at Renaissance.	In K-Mod, the 10% probability applies regardless of population, but closeness to other Barbarian cities can reduce the probability. BtS used the same procedure for Barbarians as for AI civs, just with 10% increased probability. There were some fairness clauses to prevent razing in the early game.

Rationale	Closeness shouldn't matter; different Barbarian cities aren't supposed to cooperate. Want razing only for flavor reasons; balance-wise, losing a city to Barbarians temporarily is already big setback.
Config	`RazeCityProb` is in `Civ4LeaderHeadInfos.xml`.

Decreased the number of tiles per Barbarian land unit and animal on the high difficulty levels a bit.

See also	To match the increased production costs that [251](#) imposes on the civs.
301	Early Spearman fix, no spawning of units older than the previous era

AdvCiv	K-Mod/BtS
The game creates Barbarian units with resource requirements only once the Barbarians have the tech that reveals the required resources. This only affects Spearman, which now requires Bronze Working.	Only the techs for trading the resource requirements are checked, i.e. Mining for Copper or Iron in the case of Spearman.
Moreover, Barbarian units requiring a resource can only appear on continents where at least one civ has access to that resource (or where a Barbarian city has access to it).	The same Barbarian units appear on all continents, and the game only checks if the Barbarians could work the necessary resources, not if they're actually available anywhere. (Until patch 2.13, Barbarian Spearmen didn't appear at all.)

Rationale	Early Barbarian Spearmen were probably not intended by the BtS developers. The Bronze Working requirement should make Chariots more useful against Raging Barbarians. Barbarian access to resources that don't exist on a continent is highly implausible. It's also a game balance problem when starting in the Classical or Medieval era. In BtS, the Barbarians then immediately get horse, bronze and iron units while it usually takes the AI civs some 50 turns to hook up a military resource.
See also	[CFC post](#) explaining that I didn't get the implementation of this change right until AdvCiv 1.0.

AdvCiv	K-Mod/BtS
The game only creates Barbarian units from the Barbarians' current tech era or the previous era, i.e. no Warriors and Archers in the Medieval era. Spearman is treated as a Classical-era unit because of its resource requirement.	The unit to be created is chosen uniformly at random from among the allowed units, including those from all earlier eras. Obsoletion isn't checked.

Rationale	Ancient Barbarian units become a pointless distraction at some point.
302	Tech diffusion only from civs sharing an area with Barbarians

Barbarians get research only from civs they share a landmass with. The specific conditions (for a shared landmass) change as the game progresses. For the first 100 turns, any Barbarian land unit on a landmass with a civ city suffices (or a ship on the surrounding coast), later on, a single Barbarian city isn't necessarily enough. See `CvTeam.cpp` for details. The intention is that research of New World Barbarians stops once they're driven out of the Old World. Update (v0.93): Barbarian research is now only slowed down when they don't share a landmass with a civ. (Because Longbowman is arguably a better representation of American Indians than Archer.)	Barbarians receive research based on the ratio of civs alive that have a given tech to the total number of civs alive.

Rationale	Want the research of New World Barbarians to stop once they're (nearly) eliminated in the Old World. That way, the New World will usually only have Classical-era Barbarian units, and occasionally Medieval.
See also	300: Barbarian aggressiveness now escalates based only on local cities (not those on other landmasses) kekm.28 grants Barbarians some initial tech in Advanced Start games.
Tbd.	If I really want New World Barbarians to have Longbows, I could give Barbarians more original research capacity (they already have a little, maybe by accident) and give the Barbarian leader a flavor value that matches the flavor of Feudalism.
303	Never build culture, can't build cultural buildings, found value considers only inner ring
See also	003n simplifies some AI code for Barbarians, e.g. just sets espionage and culture weights to 0.

Barbarians never build culture. Can only build the following buildings: Barracks, Granary, Lighthouse, Walls, Forge, Stable, Aqueduct, Colosseum and Bunker.	Can build all mundane buildings except: religious and research buildings, Theater, Security Bureau, Int. Agency and Broadcast Tower. Can expand its borders through Monument (or Castle); not sure how often they actually do that. The K-Mod Barbarians immediately expand their borders by building culture if they have the Music tech.

Rationale	Want each Barbarian city to appear as an independent entity. If they expand their borders as in K-Mod, the borders can easily touch, and then Barbarian cities appear like a unified empire. Aqueduct, Colosseum: Want Barbarians to occasionally have populous cities (think Tenochtitlan); for that they need all the health and happiness they can get.
See also	Important for the New World because change 300 places Barbarian cities more densely on uninhabited continents. 300 also prevents Barbarians from building a road network.
Config	Buildings are disabled through `Civ4CivilizationInfos.xml`.

Only the inner ring is taken into account when placing Barbarian cities. And some minor adjustments to the computation of Barbarian found values.	All 21 tiles in the city radius are considered. K-Mod: There is already some special code for Barbarian found values.

Rationale	Since Barbarian borders never expand, it doesn't make sense to consider the outer ring. I also like that this makes Barbarian city sites distinct from normal city sites, and a bit weaker overall.
See also	[300](#) fixes Barbarian city placement bugs.

Barbarian city placement treats all resources as having equal trade value.	Resource evaluation as if the Barbarians were a proper civilization, i.e. high value for strategic resources like Iron or Marble.

Rationale	Barbarians blocking strategic resources isn't good for gameplay: Civs that lack Iron aren't in a good position to conquer an Iron source claimed by the Barbarians. Barbarians near Marble are strange because the Barbarians obviously won't use that resource.
See also	[CFC post](#) with an attached screenshot that shows a Barbarian city with Marble as the only resource in the city radius.

304	Changes to prevent concentrated, predictable placement of Barbarians
`synchRandPlot` rewritten: If no maximal number of trials is given (`iTimeout` parameter), then `CvMap::syncRandPlot` is guaranteed to return a tile when there is at least one satisfying the selection criteria. `syncRandPlot` then returns the number of valid tiles as well.	Picks tiles at random until it finds one that satisfies the criteria, or gives up after 100 trials.
Rationale	Don't want to place a Barbarian unit on every turn (or every other turn) in the same tile, so a probabilistic failure to return a tile is good. However, the probability for failure is too high in BtS and shouldn't depend on the map dimensions – if a chance of failure is desired (it is; see 3rd blue box below), then it should be based on the proportion of valid tiles in the target area, not world-wide.
Pass a tile weight distribution to `synchRandPlot` (and to `Shelf::randomPlot`) that gets used for giving each valid tile a probability equal to its weight relative to the sum of all weights. (NB: None of the changes dependent on placement weights apply to Animals.)	Chooses uniformly at random from among the valid tiles; no weights. To make tiles with particular characteristics less likely, one can only re-roll them a fixed number of times.
See also	Change 300 uses this mechanism to make Barbarian placement in nonarable tiles less likely. Had used two re-rolls until AdvCiv 1.0, which had the (undesirable) effect of making placement on arable tiles very unlikely so long as the portion of nonarable tiles among all valid tiles was small – and not unlikely enough when the portion of nonarable tiles was large. For example, when 30% of the valid tiles are nonarable, the chance rolling a nonarable tile three times in a row is only 2.7%. If 70% are nonarable, the chance of three successes is 34.3%. With the weighted choice method, if we give arable tiles 5 times greater weight than nonarable tiles, we get a 7.9% chance of choosing a nonarable tile from among 30% and a 31.8% chance from among 70%.
Rationale	Also needed for the change described below ...
Barbarian units are less likely to appear near tiles where a Barbarian unit was recently killed or created.	All valid tiles have the same probability of spawning a Barbarian unit; the mechanism is memoryless.
Rationale	Mostly to avoid excessive and predictable Barbarian activity in a small unguarded region of the map. Can be too difficult to get such situations under control (for human and AI civs) and it feels unnatural.

See also	[312](#) provides GG points from Barbarians and thus makes XP farming a potential game balance issue.
Tbd.	Arguably, the target number of Barbarians for each continent (or shelf) should also be adjusted based on recent kills. That is, if Barbarians get killed off quickly, the creation rate should decrease. The change above only shifts the Barbarian activity around within a continent.
When only a small number of tiles remain where a Barbarian unit could be placed, then the game may probabilistically not place that unit. The probability is based on the number of eligible tiles, the target number of Barbarian units on the continent (upper limit) and on the relative placement weight (see above) of a tile chosen at random from among the eligible tiles.	The `iTimeout` parameter in BtS has a similar effect – of causing Barbarian placement to be skipped when few eligible tiles remain.
Rationale	Some mechanism like that is needed, otherwise, Barbarian (re-)appearance can become too predictable.
Patrolling Barbarians take try to avoid (to an extent) tiles with a small placement weight, i.e. tiles with poor yields or where Barbarians were recently killed or placed. The placement weight is also taken into account for choosing a target city to attack.	Terrain does not affect patrolling Barbarians. They mainly steer clear of owned tiles. Target cities are chosen through the same procedure that AI civs use, i.e. based on how well-developed the city is and how nearby. Strength memory (about garrisons spotted when the city was last visible to the Barbarians) is also taken into account.
Rationale	Just to reinforce the changes to Barbarian placement a little.
See also	[102](#) makes patrolling units (including non-Animal Barbarians) more likely to move in a consistent direction. [158](#): Changes to AI strength memory
Barbarian cities are slightly less likely to occur on landmasses where Barbarian cities were destroyed previously.	Barbarian cities can appear again and again on landmasses that civs don't want to settle.
305	Barbarian Work Boats
Barbarian cities can build Work Boats, and tend to build them early, i.e. after producing 1 or 2 military units. Build Workers a bit later than in BtS, especially in coastal cities.	Can't build Work Boats. Usually build Worker after 15 turns.
Rationale	To allow Barbarian cities to grow more population. The original developers had perhaps been worried that Work Boats would distract Barbarian cities too much from building military units. To make up for that distraction, I have Barbarian cities build Workers later.
306	Units spawned on Galleys, Barbarian naval AI
See also	The [Mongoose SDK](#) PirateMod works similarly but hasn't been merged; I only learned about this mod component after implementing my own changes. [905a](#) increases the speed of Galley but keeps Barbarian Galleys at 2 moves through the Disorganized promotion.

Barbarian land units can be placed aboard ships in the fog of far. If there is no such ship, the land units are placed on land tiles instead; the total number of Barbarians placed is the same either way. Barbarian ships with cargo target a nearby city with a naval assault, though the units can also be dropped along the way to the target. Once unloaded, the ships switch back from assault mode to "attack" mode, which works as in BtS: Harass a city for a while, then move on and patrol. Ships on patrol seek out unobserved tiles where they can receive new cargo.	Barbarian land units can only be placed on land tiles, and their AI does not allow them to enter cargo units. Created cargo units only harass cities and patrol. Units produced in Barbarian cities, however, can undertake naval assaults, which is why naval assaults do happen in BtS, but very rarely.
Rationale	Should make Barbarian sea units more interesting and harder to ignore; allows them to interact with civs that don't have worked sea tiles.
Tbd.	When a Barbarian ship has been on the map for a long period of time, say, 20 turns, it should stop patrolling and just blockade a city; can get annoying if a player just can't build a ship for lack of a coastal city.
See also	102 makes AI units on patrol more likely to move in a consistent direction.
Damaged Barbarian units gradually heal by resting for a turn from time to time (probabilistically).	Apart from hitpoints received from a received promotion, Barbarian units heal only in cities or when they can heal fully in one turn. Near civ cities, badly damaged Barbarian units never attempt to heal. If they're too weak to make an attack, they may patrol indefinitely.
Rationale	If the civs fail to finish a Barbarian unit off, it should eventually do the smart thing and heal.
Reduced the cost of Metal Casting (MC), increased the cost of Optics and (a tiny bit) Machinery.	MC is almost 4 times as expensive as its prerequisite Bronze Working and about two thirds as expensive as Machinery, the tech that MC leads to. In Warlords, Trireme was at Sailing with Bronze Working (BW) as an additional requirement, BtS moved Trireme to Metal Casting.
See also	CFC discussion The Close to Home multiplayer mod also reduces the cost of Metal Casting.

Rationale	To make Trireme easier to access. Also to make Metal Casting more attractive to research (as opposed to discovered via Great Person or Oracle) in general; earlier access to an Engineer via Forge is also a plus.
	Can't reduce the cost drastically because that area of the tech tree is quite sparse. Specifically, I don't want to shorten the path to Machinery (a Medieval-era tech) too much. The path to Optics (via Machinery) is also problematic. Machinery is already an expensive tech; can't just shift beakers there from MC. Optics had some leeway for a cost increase though.
	Don't want to move Trireme to BW because that tech already does too much (until such a time that Slavery is moved away; cf. 010/ wip section) and because rams were an essential feature of triremes and did require exactly what the MC tech represents: casting molds that weren't developed until the first millennium BC. Converting MC into a copper smelting tech would be too big a change at this time (and, if big changes ever become a consideration, then there might still be better ways to nerf BW).

307	Post-Medieval Barbarians
Barbarian units can be created by the game in all eras. Barbarians can receive and train Musketman, ~~Cavalry, Anti-Tank and SAM Infantry~~. Cannot train Rifleman, Grenadier, Paratrooper, Frigate. As of **v0.94**, Musketman is the only post-Medieval unit that Barbarians can receive or train.	Can only receive and train Warrior, Archer, Spearman, Axeman, Swordsman, Horse Archer, Longbow, Maceman, Galley. No Barbarian units created by the game in Renaissance and beyond. Can train Rifleman, Grenadier and Frigate.
Barbarians can't receive cities in Renaissance and beyond.	Barbarian cities still appear in the Industrial era.
Starting from Renaissance, Barbarian research ignores tech prerequisites.

The Raging Barbarians option creates more Barbarian units in later eras: the tiles-per-unit divisor is reduced to 60% in the Ancient era, 52.5% in Classical, 45% in Medieval, 37.5% in Renaissance, 30% in Industrial, 22.5% in Modern and 15% in Future (relative to the divisor without RB). | In BtS, Barbarian research always ignores tech prereqs, i.e. Barbarians can research e.g. Pottery and the Wheel simultaneously. K-Mod has changed this, so that Barbarians only make progress on techs that they could research if they were a civ.

RB reduces the divisor to 50% in all eras. |

Rationale	Gunpowder units mostly for Terra and similar maps – on normal maps, there is typically no land left for post-Medieval Barbarians to appear on. Therefore the reduced divisor for later eras with RB; so that RB players get to see the occasional post-Medieval Barbarian.
	Muskets are supposed to represent natives that have acquired firearms (through trade or as spoils). Until v0.94, Barbarian Cavalry and (as a fairly ineffective type of infantry) Anti-Tank and SAM were also allowed. I'm reverting this because Cavalry practically never appears (requires a local Horse resource to train), and because advanced weapons like bazookas and MANPADS are jarring in the hands of the Barbarians. Would have to create custom units for the Barbarians to represent adequately e.g. mounted post-Columbian Amerindians or Daesh insurgents.
	Don't want Barbarians with firearms to be on par with properly trained armies, hence no Rifle, Grenadier and Frigate.
	No Pikeman: Two Medieval Barbarian units (Longbow, Mace) are enough, and I want mounted units to be effective against Barbarians, especially Cuirs/ Conquistadores.
	Barbarian research ignoring tech prereqs allows Barbarians to catch up quickly once colonies are founded. By the time New World Barbarians turn aggressive, they'll typically have Muskets. Interestingly, Chronis seems to have had the same idea years before me (though I haven't checked if he implemented it in the same way).
	No Barbarian cities past Renaissance: Seems more realistic, and enough cities are created in the New World during the early eras.
See also	302 stops Barbarian research if they stop sharing a continent with any civ. 301 lets Barbarians only build units that some civ on the same continent can build. kekm.6 disables Barbarian Spies.

Marines (or is it Infantry?) as free defensive units in Modern-era starts.	Machine Gun as free defensive unit.
Rationale	Can't use Machine Guns to prevent Barbarians from pillaging.
308	Tech costs adjusted to Barbarian settings

Increased tech costs for all civs in games with Raging Barbarians in the Classical in Medieval era.

Slightly decreased Ancient and Classical tech costs for all civs in game without Barbarians.

Rationale	RB forces the civs to dedicate resources into fighting Barbarians, and as a result, global research tends to lag behind the historical time line.

309	No Animals option, patrolling animals prefer their native terrain/features
AdvCiv	*BtS*
"No Animals" option added; disabled by default (i.e. animals do appear by default). "No Espionage" option removed. "No Barbarians" implies "No Animals". NB: Without animals, Barbarians tend to appear a bit quicker.	Can't disable just animals, only all Barbarians. "No Espionage" was added with patch 3.17; converts espionage points to culture. (The game counts animals as Barbarians when deciding how many Barbarians to create on a given turn. Therefore, the presence of animals slows down the placement of proper Barbarians.)
Rationale	Animals are nice for teaching beginners how combat works, but too silly for some tastes. I guess one can always imagine they're "really" human nomads, say, "lion warriors" instead of lions.

	Don't want to clutter the Custom Game screen with options. The No Espionage option wasn't properly implemented and can't be recommended in its current state. Players who dislike espionage are arguably better off just ignoring it.
Config	Can get the option back by setting `bVisible` to 1 for `GAMEOPTION_NO_ESPIONAGE` in `Assets\XML\GameInfo\CIV4GameOptionInfos.xml`.

Patrolling animals favor their native terrain and features (as defined in `Civ4UnitInfo.xml`). They still enter other tiles, but with a smaller probability.	Animals spawn only on their native terrain and features but move indiscriminately.
Rationale	Flavor
Credits	Idea from [Mongoose SDK](#) AnimalMod

Animals aren't removed from landmasses without civ cities (unless the landmass gets too crowded with units; change 300).	Once the game stops creating animals, it removes one animal per turn from each landmass.
Rationale	Buffalo flavor for the New World. Once colonized, the animals will start disappearing.

Animals may peacefully enter unowned tiles with - a resource if that resource requires a tech to be revealed or - an improvement unless it's a goody hut. May always enter unowned tiles through an attack (no change).	Unless an animal attacks, it can't enter unowned tiles with resources or improvements.
Rationale	The resource avoidance can give away hidden resources; no fun to keep track of this. Could simply always allow animals to enter resources – it's not like this prevents players from settling near those resources –, but predators coexisting peacefully with herbivorous resources could seem strange to some players. Improvement: I don't think this ever occurs, but animals reclaiming abandoned structures could be flavorful. Don't want them to block goody huts though.
310	Great Wall reworked, balance changes to Great Lighthouse and Colossus
Config	Can be reverted in `CIV4BuildingInfos.xml`. The abilities that require a game option are handled by the DLL. That can be disabled in XML by setting the `bConditionalAbilities` element to 0 for the Great Wall. Then the Great Wall will always have whichever abilities are set in XML.

The Great Wall (TGW): cost 300, req. Archery, 2 Walls (1 on Duel and Tiny world size, 3 on Huge), obs. with Corporation, 2 Great Merchant points, 4 culture, abilities based on game settings: i. +1 trade route in cities on the same continent except when playing with Raging Barbarians (RB). This ability is based on a "global trade routes" ability that was already in the game but unused. Now restricted to cities on the same continent. ii. Prevents Barbarians from entering your	Cost 150, Masonry, no prereq. buildings, no obsoletion, 2 Great Spy points, [Great Engineer in Warlords] 2 culture, abilities (ii) and (iii) regardless of settings.

borders on this continent (no change) except when playing with No Barbarians. iii. +100% emergence of Great Generals within owner's borders (no change) if playing with RB or No Barbarians. I.e. TGW doesn't have this ability anymore under normal settings. Can be built on Classical and earlier starts.	Can be built on Ancient and earlier starts.

Rationale	A strangely cheap wonder in BtS considering that it represents (arguably) the most massive building on Earth. Can be costlier now that Barbarian activity peaks later (and OK to allow it with Classical era start). Removing the GG ability under normal settings keeps the rules complexity in check (wouldn't be a problem wrt. game balance). Trade route flavor: One purpose of the Chinese Great Wall was to protect traders. Some segments protected a portion of the Silk Road. Similar to Castles providing an extra trade route. GM points go along with this, but the goal is also to make the espionage system easier to avoid for players who wish to avoid it. A nice side-effect of the Wall prereq. is that Protective leaders get an implicit discount on TGW. Also makes sure that TGW can't be built before founding a second city – players should analyze whether they're exposed to Barbarian attacks before deciding to build TGW. The Archery req. is to make Archery a bit better, and to make Masonry less cluttered visually; TGW still requires Masonry for the prereq. Walls. I believe this makes some sense historically as well – walls (with protruding towers) are much more useful with effective projectiles. (But this doesn't explain why ordinary Walls don't require Archery.) Culture rate: Don't want it to be curiously small, but also don't want to speed up border expansion much because, as soon as that happens, the Great Wall graphic stops aligning with the cultural borders.
See also	Without [140](#), which changes the formulas for the number of prerequisite buildings, only 1 Wall would be required on Standard-size maps. [131c](#) decreases the military power value of TGW. [911a](#) adds a Spy specialist slot to Courthouse in order to enable an early Great Spy.
Tbd.	I'd prefer Barbarian naval landings not being subject to the Great Wall; but I think this would be rather complicated to change as it involves pathfinding code. A start would be to search the codebase for all instances of "BorderObstacle".

The 3D graphic for the Great Wall on the main map is not shown along borders with other civs and not along unhabitable tiles. Tiles that are only adjacent to unhabitable tiles and border tiles are also excluded.	Along the full length of the owner's cultural borders on the same landmass as the Great Wall city except at the coast.

Config	`GREAT_WALL_GRAPHIC_MODE` in `GlobalDefines_advc.xml`
Rationale	If the Great Wall isn't constructed in the early game, the graphic can get very long in BtS. Since the (primary effect of) the wall helps only against Barbarians, which don't normally enter from the territory of another civ, it makes sense to me to omit civ-to-civ borders. If this means that no wall segments are shown at all, then that's also OK with me. The last condition for placing a wall segment is supposed to prevent Gaza-like strips of tiles enclosed between Great Wall and some other type of frontier or obstacle.
See also	"Habitable" tiles are tiles with 0 nature yield when hill yield changes are ignored; same

	as for Barbarian placement ([300](#)).
Credits	I took the idea for the implementation from Leoreth ([link](#)). I also took a look at the Dawn of Civilization [code](#), but it wasn't easily portable.

The Great Lighthouse (GLH): cost 250; obs. with Astronomy	cost 200 in BtS, 300 in K-Mod; obs. with Corporation
Rationale	Too powerful in BtS, at least compared with most other wonders. And I don't like that this ancient wonder becomes most effective in renaissance (with Astronomy). I meant to nerf this later on (along with other balance changes), but with TGW obsolete at Corporation, GLH obsoletion needs to be moved right away.

The Colossus: cost 250 (as in K-Mod); obs. with Chemistry.	cost 150 in BtS, 250 in K-Mod; obs. with Astronomy.
Rationale	Don't want both Colossus and GLH obsolete at Astronomy. The K-Mod cost increase seems reasonable; it's an extraordinarily cheap wonder in BtS.
Tbd.	Perhaps Forge should be nerfed instead – if Forges weren't spammed everywhere, Colossus wouldn't be such a small investment.

Versailles generates Great Spy points	Great Merchant points
Rationale	Compensation for the change to the Great Wall. Rather symbolic as Versailles is a much later and less useful wonder than the Great Wall.

311	Uprising events
AdvCiv	*BtS*
Place 2 units in the Ancient era, 3 in the Classical era, 4 in the Medieval era. The era of the player for whom the event has triggered counts.	1 to 6 units depending on the map size; era doesn't matter.
Rationale	4 Vedic Archers on a Standard-size map is too many. And of course the map size should have no bearing on the number of units.
See also	My [initial formula](#) had created 5 units in the Medieval era. That was still too many it appears (and, on a Standard-size map, it's actually more than in BtS): [CFC post](#)
The units can only be placed on tiles that are exactly two tiles away from the borders of the player for whom the event has triggered.	Adjacent to that player's borders.
Rationale	To give the player more time to rally troops. Ideally, a player with a reasonable military should never lose a city to an uprising event (losing one or several defending units is already more punishing than most contemporary events).
See also	The CFC post linked above.
Vedic Aryans can't trigger for any player until at least one civ knows Priesthood.	Polytheism
Credits	From BUFFY version 003.

312	XP from Barbarians counts half for Great General
XP from combat against Barbarians counts 50%, rounded down, toward Great Generals (GG). Rounding down means that 1 XP (i.e. after combat with very one-sided odds or withdrawal)	XP from Barbarian combat never counts for GG.

doesn't count at all toward GG. Units that already have 10 XP can't gain further XP from Barbarians (no change), and thus can't contribute to a GG either.	
XP multiplier for attacking Barbarians reduced to 3; no change when defending against Barbarians. Thus, can't hope to gain more than 1 XP (0 GG points) by fighting a Barbarian unit at odds above 90%.	XP multipliers are 4 when attacking and 2 when defending. Can expect to gain 2 XP from attacking a Barbarian unit even at 95% odds.
Upper limit for XP gain per combat set to 6. Consequently, can gain at most 3 GG points per Barbarian combat.	Can gain up to 10 XP from a single attack.

Rationale	As an extra incentive for fighting Barbarians rather than (completely) preventing them from being placed through fogbusting. It's also not plausible that wars against Barbarians should produce no GG. Ancient China had plenty of GG that only fought against peoples that Civ would represent as Barbarians. I'd like to give players a (fighting) chance to get a GG without starting a war.
	The original developers must've been worried about Barbarian XP farming, and for good reason. Having Barbarians respawn in some cul-de-sac isn't difficult. Can be worthwhile to exploit just for getting units to 10 XP; which is why I think some restrictions were needed regardless of GG points.
Tbd.	Farming may still be feasible; is a 15% risk of losing a unit worth 2 XP and 1 GG point? Could be, especially if it's an outdated unit … Reduce upper bound for per-combat XP to 5 (2 GG points)?
	Perhaps apply the XP restrictions to all combat; would like to reduce combat XP in general in order to lower the stakes in combat.
See also	304 makes Barbarian units less likely to be placed in tiles where Barbarian units were previously created or destroyed. This should at least prevent XP farming from having any major impact on the game balance.

313	Difficulty-based combat bonuses vs. Barbarians
Set the Barbarian combat penalty against humans to 5% on Monarch, Emperor and Immortal. This is in addition to the 10% penalty for Barbarian Galleys from K-Mod ("Disorganized" promotion). No change to the other difficulty settings and no change to the Barbarian penalties against the AI.	The penalty is 40% on Settler and drops by 10 percentage points with each difficulty level to 10% on Noble. Then 5% on Prince, and 0 on all levels above Prince.

Rationale	Despite the nerf to fogbusting (300), it still seemed best to rely mostly on fogbusting on the higher difficulty settings because fighting Barbarians at tech parity is very costly. This slight change to the combat penalty should make a significant difference because, when units are evenly matched, even a slight change in combat strength tips the combat odds from 50:50 to 60:40 or more.
	I haven't changed Deity because degenerate tactics (such as excessive fogbusting) are normal on that difficulty. Also matches the degression of the animal penalty, which reaches 0 only on Deity (see below).

The combat penalty that animals receive is:					
difficulty	*vs. human*	*vs. AI*	*difficulty*	*vs. human*	*vs. AI*
Settler	-60% (-100%)	-25% (-50%) on all difficulty settings	Settler	-70% (-110%)	-40% (-65%) on all difficulty settings
Chieftain	-45% (-75%)		Chieftain	-60% (-90%)	
Warlord	-30% (-50%)		Warlord	-50% (-70%)	
Noble	-25% (-35%)		Noble	-40% (-50%)	
Prince	-20% (-25%)		Prince	-30% (-35%)	
Monarch	-15% (-20%)		Monarch	-20% (-25%)	
Emperor	-10% (-15%)		Emperor	-10% (-15%)	
Immortal	-5% (-10%)		Immortal	-5% (-10%)	
Deity	0% (0%)		Deity	0% (0%)	
The numbers in parentheses include the penalty for Barbarians, which, as in BtS, also applies to Animals.					
Rationale	The total penalty against AI units and the penalties against humans on the medium difficulty settings seemed needlessly high to me. Losing an exploration unit now and then won't hurt the AI much and could help delay the first contact between far-flung civs.				

314	Tribal villages ("goody huts") revised
See also	315b makes it easier for Explorers to enter guarded Tribal Villages.
Config	Most of the specific changes to the Tribal Village outcomes are customizable and revertible through `Civ4GoodyInfo.xml`. I haven't added any new tags; instead, I'm e.g. using the Gold and Tech tags together to represent tech progress.
Tbd.	Would be better to add new tags. Not so painful anymore with advc.tag and 003t.

AdvCiv	*Vanilla Civ 4* (no changes in WL/BtS/K-Mod)
All effects of goody huts are adjusted to the game progress. No such adjustment during the first 50 turns, nor after turn 250 (on Normal speed; these turn numbers are speed-adjusted). Most of the adjustments happen through a multiplier that is computed from the current game turn. That multiplier starts at 1 on turn 50 and increases superlinearly (power law) to 10 on turn 250 and then stays 10 for the rest of the game.	The Warrior outcome is blocked until 20 turns have passed (not speed-adjusted), and Barbarians can only appear at a certain distance from cities. Apart from that, only the Tech outcome somewhat scales with the game progress (by granting a tech that can currently be researched).

Rationale	The main goal is to make huts discovered via Caravels or Galleons more rewarding, and thus also give Explorers a better use. And some of the free techs granted in BtS are too powerful in the early game – I guess that's why players commonly play without goody huts.
	I'm tying the adjustment factor to the game turn rather than e.g. era because I don't want players to wait for an era transition before entering a goody hut. That's also the reason for the flat effects during the first 50 turns. Gold doesn't have any use in the very early game, so players might decide to let a hut "grow" before entering it.
	The power-law function is supposed to mirror the overall economic growth, which is superlinear. I'm freezing the effects on turn 250 because I don't think that industrial civilizations can gain that much from visiting or raiding Bronze Age (or Chalcolithic) villages. "Times 10" is a nice and simple maximal effect (and one can kind of tell that the maximum is reached when a multiple of 10 gold is payed out).
Config	Start turn, peak turn and maximal multiplier are set in `GlobalDefines_advc.xml`.

The High gold outcome grants between 40 and 100 gold, and the speed adjustment is based on the training cost modifier, i.e. only 200% on Marathon. Other than that, the payout during the first 50 turns is as in BtS. By turn 250, the payout is multiplied by ten (after being multiplied by the training cost modifier).	The Low gold outcome grants between 20 and 60 gold and High gold is between 20 and 120. Low is the more likely outcome on the medium and high difficulty settings (no change); e.g. on Monarch, Low is four times as likely as High. The amount of gold is fully adjusted to game speed, i.e. tripled on Marathon.

Rationale	120 gold is too much early on, but expected values between 40 and 50 are OK I think, considering that gold can't be utilized until city maintenance becomes a factor, whereas e.g. a free Scout worth 15 production (=30 gold?) is immediately useful.
	Marathon games tend to be played on larger maps with more huts per civ, hence the smaller speed-adjustment factor than in BtS.

During the first 50 turns, goody huts grant 35 to 75 progress toward a tech. The tech is directly discovered only if that progress is enough to cover at least 80% of the (remaining) tech cost. The granted progress is adjusted to the game speed and game turn in the same way as gold (see above). The tech is chosen uniformly at random from all pre-Industrial techs that the civ entering the Village is able to research, including the currently researched tech.	When the Tech outcome is rolled, the hut grants a free tech regardless of that tech's cost. The tech is chosen uniformly at random from among those currently researchable techs that are flagged as `bGoody` in `Civ4TechInfos.xml`. No game speed or progress adjustment, but tech costs increase with the game speed and progress. That said, the only post-Classical `bGoody` techs are Music and Astronomy, so the only way to benefit significantly from huts across the ocean is by reaching them in between Optics and Astronomy, and even then it's a long shot. Note about a K-Mod change (from the summary in the K-Mod thread): "*Free technology bonuses from goody huts are now allowed without settling a city first.*"

Rationale	The expected amount of research is about the same as the expected amount of gold (considering that some research can go to waste when the tech is cheap or already nearly discovered). In the early game, research is going to be more valuable because gold can't be spent immediately; in Renaissance, gold is going to be more valuable because civs can funnel it into a tech of their choice via the research slider.

I guess the `bGoody` techs were chosen based on whether a "primitive" civilization could plausibly possess them. I don't think this works for Astronomy, which unlocks Galleon and Observatory, nor for most other post-Ancient techs. Tech progress somewhat gets around this plausibility issue by merely suggesting that the primitives contribute to the discovery, maybe like modern astronomers employing knowledge from ancient Babylonian records. Such contributions strain credibility for techs like Radio, so I've taken the Industrial and Modern techs off limits. |
| *Config* | `bAllGoodyTechs` flag in `Civ4EraInfos.xml`. If that flag is not set, then techs from the era can only be obtained from a goody hut if they're set to `bGoody` in `Civ4TechInfo.xml`. I.e. it's also possible to revert to the BtS system by setting `bAllGoodyTechs` to `0` for all eras. |
| *Tbd.* | Should perhaps also adjust the outcome to the map size. Tech costs are higher on larger maps (e.g. 100% on Duel, 130% on Standard), and the current payout is going to grant a full tech only rarely on large maps, particularly on high difficulty settings. Then again, one could argue that larger maps have more huts and therefore each hut should provide less research than on smaller maps, if anything.

Might also want to adjust to game era; see 910. |

Outcomes that can't be scaled up in a straightforward way, namely Map, XP, Heal and Scout, get a chance of being "upgraded" if they occur later than turn 50. The upgrade probability reaches 100% around turn 160. For Map, XP and Heal, an upgrade means that another positive outcome is chosen at random and both are applied. For the Scout outcome, the upgrade is a free Worker instead of the free Scout.	Regardless of the game progress, the Map outcome reveals a randomized subset of tiles within a radius of 4, XP grants 5 XP points, Heal heals the unit entering the Village and Scout grants a free Scout.

Rationale	These four are pretty worthless when triggered by an Explorer; can't have so many dud outcomes. A higher amount of XP wouldn't help (not needed on an Explorer) and would be too similar to a Great Warlord.

A free Scout seems out of place when that unit is already obsolete; better to replace that outcome entirely than to roll an additional one. In Renaissance, a free Worker is not as valuable as tech progress or gold, but it should work well enough as a consolation prize. |

The Warrior outcome produces a free unit chosen based on combat strength and randomness from among those pre-Industrial combat units that don't require resources and for which the Barbarians have all the prerequisite techs, i.e. Warrior, Archer, Longbow or Musketman. (The same procedure is used when playing with "No Barbarians" as that option does not prevent the Barbarians from discovering tech.)	Always produces a Warrior.
The types of hostile units (Barbarians outcome) are also chosen as above. The lower bound for the number of hostile units (`iMinBarbarians`) increases a little over time, but there's also an upper bound of 2 + `iMinBarbarians`.	Hostiles are always Warriors. The Weak Barbarians outcome usually spawns 1 or 2 Warriors and Strong 2 to 4, though as many as 8 are possible.
A non-hostile free unit has a chance of receiving up to two free promotions. Each of the two promotions is assigned with a probability equal to half the upgrade probability (see Scout above). If a promotion is assigned, the specific promotion is chosen randomly from Combat1, Guerilla1, Guerilla2, Woodsman1, Woodsman2 and Cover. The surrounding terrain also factors into the selection of the promotion.	The free Warrior starts with 0 XP and without any promotions.
Rationale	Choosing the unit based on the game era would be simpler, but then Barbarian Archers would appear too early (even if game era minus 1 was used). In Renaissance, the received unit will usually have to be upgraded in order to be useful; the free promotions should make such an upgrade worth considering. I've picked promotions with a "native" flavor. The selection algorithm is pretty complicated, but I don't think players need to worry about that.
See also	302 curbs Barbarian research when civs don't share a continent with Barbarians. (But this doesn't apply when playing with "No Barbarians".)
The Worker outcome can't occur until turn 20 (on Normal speed). Halved the probability of the Settler outcome on Chieftain and Settler difficulty and set it to 0 on Warlord.	Only the Warrior outcome has such a restriction, presumably to prevent a super early rush. The Worker outcome can't occur on Prince difficulty and higher (no change).
Rationale	I think novice players should play on Noble difficulty, but, for a R&F game, it could make sense to go down to Warlord, so that difficulty setting shouldn't have grossly unbalanced goodies.
Config	The Settler changes are implemented through `CivHandicapInfo.xml`.
Increased the probability of the Tech outcome at the expense of the Gold outcome so that Gold is less than two times as likely as Tech.	For most difficulty settings, Gold is three times as likely as Tech.
Rationale	The Tech outcome is more interesting than Gold. Now that their power level is similar, Tech can be awarded more frequently.
Config	Through `Civ4HandicapInfo.xml`
Increased the size of the Tribal Village 3D model on the map.	

Rationale	They're hard to spot on Forest tiles. I've also tried a more reddish, saturated color, but the increased size seems sufficient and a color change alone doesn't help because the BtS-size huts barely poke out of the trees.
See also	By now, I've added a Tribal Village "bubble" to the Resource layer (change 004z), so the size of the 3D model isn't so relevant anymore.
Config	Through `XML\ArtCiv4ArtDefines_Improvement.xml`

Tribal villages are placed when starting in the Ancient, Classical or Medieval era (unless diabled through game options).	Only when starting in the Ancient or Classical era.
Rationale	Now that the effects scale, I don't see a reason to place huts in the Classical era but not in the Medieval era. In both cases, the human player starts with a single exploration unit. I like the huts as rewards for discovering unclaimed continents. When starting in Renaissance though, this is too easy as Caravels are available from the beginning.
The minimal distance between two Tribal Villages gets adjusted to the feature grain value of the map size.	Tribal villages need to be at least 4 tiles apart. The feature grain value depends on the map size and affects the size of clusters of terrain features. It's -1 for Duel and Tiny size and +1 for Large and Huge size. It doesn't affect Tribal Villages, which technically are terrain improvements.
Rationale	Space them out more when there is more space. Didn't want to add an improvement grain value just for this purpose.
Config	The base value for the minimal distance between Tribal Villages is set through `iGoodyRange` in `Civ4ImprovementInfos.xml`, the feature grain in `Civ4WorldInfos.xml`. AdvCiv doesn't modify any of those values.

315	Recon units
Rationale	To make these units more useful. Scout has the problem that Warrior provides sufficient exploration early on while also being highly useful for military happiness, city defense and for staving off Barbarians. Moreover, on high difficulty settings, the AI grabs most of the goody huts. Explorer is rarely used at all.
315a	Scout can attack Animals.
Rationale	Makes it easier to get promotions. Plus Scout requires Hunting, so it should know how to hunt.
Config	I've added a new tag `bOnlyAttackAnimals` to `Civ4UnitInfos.xml`.
Tbd.	Give Scout the ability to enter Peaks. Will probably have to adjust the global `pathDestValid` function. Apart from that, I've already gone through `isImpassable` checks in the code – should be OK. Might want to go through `isPeak` checks as well. Since 030, the AI code doesn't 100% support the ability to cross peaks that separate two land areas. This doesn't apply to any of the AI routines relevant for recon units though; should be fully functional.
See also	Change 500b, once enabled, will make Warrior less useful for military happiness. 912c already reduces the military happiness from Hereditary Rule. 124 (trade routes only through revealed tiles) and the increased AI aggressiveness of K-Mod/UWAI compared with BtS reward repeated exploration. 250e reduces the number of free AI Scouts on Emperor difficulty.

315b	Explorer can attack Barbarians, but can't capture cities and gets no free promotions. Culture garrison value: 3 (0 in BtS; Warrior 3, Archer 4, Axeman 4)
Rationale	It's quite common for goody huts to be guarded by a Barbarian unit by the time Explorer becomes available. In BtS, there is no way to enter such goody huts with an explorer. That's kind of realistic – one can't expect friendly relations with the villagers after killing their guards –, but the Explorer needs a buff, and it should be a buff that makes it better at exploring rather than giving the Explorer some auxiliary use in warfare. It's also wrong to imply that the European explorers engaging in combat with American Indians always acted in self-defense.
	With "no capture" as an additional ability ("attack only Barbarians" merely replaces "can only defend"), the unit gets too complicated, or at least too verbose, so I'm removing the promotions. They're not really useful anyway because Barbarians rarely attack Explorers, at least not on Forest, Jungle or Hill tiles, and Guerilla2 and Woodsman2 are unattractive for a unit that already ignores terrain movement costs.
	Culture garrison value: E.g. Columbus became governor of Hispaniola. It's a pretty low value, but the same value as Axeman seems a bit high.
Config	I've added a new tag `bOnlyAttackBarbarians` to `Civ4UnitInfos.xml`.
Tbd.	Replace the "ignore terrain movement cost" ability with a free Sentry promotion. (Does that work inside a Caravel? Should it?) When Scout gets the ability to enter Peaks, Explorer will have to get it too.
	Are huts sometimes guarded by Archers? Even if an Explorer starts with a promotion from Barracks, a fully fortified Archer is a tough opponent ...
See also	314 makes overseas goody huts discovered in Renaissance more rewarding.

AdvCiv	*BtS*
The move-all-terrain ability prevents Gunship from capturing Workers. Explorer can capture Barbarian Workers.	The "cannot capture cities" ability prevents Gunship (and any other unit that can't capture cities) from capturing Workers.

See also	Not relevant for now because 010 currently causes captured Workers to be destroyed.
Rationale	I'd like to enable Worker stealing again somehow in the future, and then another (rare) use for Explorer could be to steal Barbarian Workers for nearby colonies.
315c	Scout gets +100% combat strength against all Barbarians, not just against Animals. The AI may use Scouts and other exploration units to guard city sites (i.e. for fogbusting) when there is nothing to explore.
Rationale	Warrior is still the better unit against Barbarians (as it should be) in every regard except mobility: greater benefit from combat promotions due to greater base strength; can attack pillagers; +25% city defense.
315d	Cap on early Scouts from huts

Recon units entering a hut can't get the Scout outcome if their owner already has two or more exploration units.	Scouts can beget more Scouts, and, in the first 20 or so turns, it's not so rare to end up with a total of three Scouts when starting with a Scout. Here someone reported three Scouts in a row for a total of four.

Rationale	The third Scout is a weak outcome, and no fun because the map gets explored too fast.
315e	Huts within 3 tiles of the edge of a non-wrapping map can't yield the reveal-map outcome.

Rationale	The reveal-map outcome is weak in any case. When one direction is blocked, it gets too weak I think.

500	Dormant changes; to be enabled or discarded sometime in the future.	
See also	098: The changes to the doubling of building culture rates would also fit here.	
AdvCiv		*BtS*
500a	Land of two rivers	
Tiles bordering on more than one river get twice the commerce yield from rivers, i.e. 2 commerce instead of 1. A shortcut on a far-winding river also works. Normally, only a few tiles per map qualify. Exact condition: two river segments that aren't connected along the tile in question. Settling on a double-river tile leads to 2 commerce on the city tile (akin to settling on plains hill).		A tile is either river-side (+1 commerce) or isn't.
Rationale	Plan to nerf the Financial trait further (908a), which will make rivers less useful. The double commerce bonus will make up for that a tiny bit. But mostly just for added flavor. Update: Probably won't ever enable this. Would provide too much extra commerce in Earth scenarios.	
Config	Disabled in `CvPlot.cpp` (`calculateNatureYield`, `isConnectRiverSegments`)	
500b	Demand better protection	
Anger from lack of protection computed based on the defensive strength of the local units and population size. A single weak unit doesn't always suffice to avoid "We demand better military protection". A city requires a total defensive strength equal to at least its population. Defensive strength is computed as combat strength increased by defensive modifiers (e.g. from Archer, Wall, Garrison promotion – but not from culture-based defense). For units that can't receive defensive modifiers, none are counted. Note that, so long as far for safety is the only cause of fractional anger in the city, i.e. in the early game, rounding will make e.g. a Warrior fully sufficient still at size 5 (not 2 or 3 as one would expect from its defensive strength of 2.5). The anger is proportional to the lacking defensive strength; e.g. an almost sufficient garrison will lead to just 1 anger. Recon and Helicopter units count if they have a positive culture garrison value (i.e. Explorer as per change 315b, Gunship).		Any one military land unit (except Recon, Helicopter; based on the `MilitaryHappiness` flag in XML) suffices to avoid "We demand military protection".

Rationale	Requiring just any military unit becomes a pretty nonsensical rule by the Medieval era. Can either abolish it, or change it so that credible garrisons are required.
	Not sure if unit health should matter. I think it would create unnecessary distraction if a city, e.g. after a Barbarian attack, would clamor for better protection for a couple of turns. Likewise, bombardment damage to city defense and fortification bonus shouldn't matter. Fear-for-safety anger isn't supposed to punish players whose cities come close to being conquered (that risk is punishment enough), it's supposed to punish careless/ precarious precautions before the city actually comes under threat.
	Counting Walls and Castle buffs those buildings a little. Counting promotions has the side benefit of encouraging promotions at peacetime.
Config	Disabled through `DEMAND_BETTER_PROTECTION` in `GlobalDefines_advc`. Can also enable a penalty for outdated units there and tweak the defensive strength target.
See also	Defensive strength is already used for revolt probabilities; see 101. However, tile defensive doesn't count for revolt probabilities and unit health does count.
	CFC post by me laying out rationales and explaining why I'm not quite happy with this change.
	The mod component Scaling Garrison Unhappiness by Imp. Knoedel uses culture garrison values instead of defensive strength. (But in my estimation, culture garrison strength increases too slowly over the course of a game, and it's another value to be displayed by the UI.)
Tbd.	Defense from obsolete buildings shouldn't count. Will have to add a data member `CvCity::m_iNonObsoleteBuildingDefense` for this because building defense is cached in `m_iBuildingDefense`.
	Perhaps the Nationalism tech should have the ability "Cities don't demand protection". Keeping garrisons at the proper strength could become annoying toward the late game. Or, alternatively, cities that don't have a non-friendly tile within a range of [how many?] tiles should not require any garrison. (Idea from CFC user Zeta Nexus)
	Need to show "Garrison strength x.x, population: y" in help text when there is anger. Otherwise it's too difficult to learn.
	Performance was slightly problematic until I disabled the check for outdated units. Should check sometime if there is still any issue now. If so, then the result of `CvCity::getNoMilitaryPercentAnger` should be cached. That cache will have to be updated once per turn (sufficient for dealing with defensive buildings and unit obsoletion); and in `CvUnit::setXY`, `CvCity::setPopulation`; and after assigning a promotion.
	I've been thinking about making unit upgrades mandatory (if not upgraded in time, units refuse to fight with outdated equipment and defect). Then this change will be important because otherwise players could try to play around unit obsoletion (e.g. don't get Hunting so that Warriors can be trained indefinitely) so that they don't have to upgrade their city garrisons.

550	Changes to tech trading
See also	130z deals with tech gifted by the AI to rivals
550a	Tech trade value dependent on tech score and power
AdvCiv	*BtS*

AI gives tech away cheaper (or expects to get it cheaper) when the receiving side is less advanced and powerful than the giving side; charges more if the receiving side is more advanced/ powerful. The effect is small in the early game and increases as the game progresses. Exception: Doesn't apply to vassal-master relationships, i.e. vassals don't expect to get tech from their master at a discount.	Tech trade value is independent of how well the involved civs are doing. (Vassals can receive free tech from an AI master if they're behind, but that's not a matter of trade value.) BBAI and K-Mod let AI civs that are falling behind technologically check for tech trades more frequently (no change in AdvCiv). Vassals charge less from their master for resources, trade embargoes and civic/religion changes. Tech trades are unaffected by vassal/ master relation.
Rationale	A little extra catch-up mechanism. Also, when dealing with a backwards civ, it's smarter to be generous than to insist on a square deal. For vassal-master, the basic idea in BtS is that the master can trade favorably with the vassal. Don't want to turn this upside down. It's important to keep the effect small in the early game because it'll otherwise make bee-lining to expensive low-utility techs like Aesthetics or Compass even more powerful on the highest difficulty levels.
550b	AI doesn't make bad offers for human tech
AdvCiv	*K-Mod*
The AI doesn't propose or counter-propose any trade to a human civ that gives tech to the AI civ and is more than 50% in favor of the AI civ. The AI always offers something when proposing a trade.	The AI can (counter-)propose trades that are up to 300% in its own favor. May offer nothing at all.
Rationale	I used to accept bad AI offers for tech because I believed that they're due to partial research progress of the AI. That can be the case but often isn't; so it's a bit of a trap.
Tbd.	The AI probably shouldn't contact players with bad non-tech deals either.
550c	Changes to tech monopoly thresholds ("don't want to trade just yet")
AdvCiv	*BtS*
AI is more willing to trade tech if it has contact with few civs and still hasn't met most civs. E.g. if an AI civ knows only one civ, the AI civ acts as if 40% of the other known civs already knew the tech (when in fact it's 0%).	AI willingness to trade a tech is based on the percentage of known civs (excluding itself) that already know the tech. If the AI knows only one civ, the AI will hardly ever trade (unless Friendly) because the percentage is either 100%, meaning the other civ already knows the tech, or 0%, meaning the AI tries to monopolize the tech.
Rationale	This should make civs on small continents fare better.

The AI leader-specific thresholds for monopolies (`TechTradeKnownPercent`; between 0 and 100) are randomly increased or decreased by up to 15 points. This random adjustment is applied once per game to each tech separately, i.e. it's not the same for all techs, and can't be manipulated through repeated trade requests by a human player.	`TechTradeKnownPercent` is only adjusted for techs that the AI wants to monopolize badly, in particular military techs. There's no random component.
Rationale	The BtS monopoly thresholds are a bit too predictable in some cases, especially when there are just two or three civs on a continent. And e.g. Pleased Tokugawa *never* trading anything useful is depressing.
Tbd.	Perhaps the issues with small continents are already taken care of by the change above; not sure if the randomization is still needed.
AI considers the unique unit and building of the prospective recipient of a tech when computing a monopoly value.	Only looks at the default units and buildings that a tech unlocks when computing the tech monopoly value.
Rationale	A small tweak. Taking into account more specific information can't hurt.
550d	Tech costs reduced if tech trading disabled
When the "No Tech Trading" option is checked on the Custom game screen, tech costs are decreased based on the default player count of the map for all eras except Ancient and Future. The research bonus is the highest for Medieval and Renaissance tech.	"No Tech Trading" doesn't affect research speed, and such games tend to lag behind the historical time line.
Rationale	A lower bonus in the early game because tech trading doesn't become available until the middle of the Classical era even if it's enabled. A lower bonus after Renaissance because tech progress tends to get more disparate the longer the games lasts, and so fewer and fewer tech trades happen.
Config	Can be adjusted or disabled through `GlobalDefines_advc.xml`.
Tbd.	Would be unnecessary if there was a proper tech diffusion system to replace tech trading.
See also	910 makes various adjustments to tech costs based on game options.
	CFC post laying out the formula for my no-tech-trading modifier: link
550e	Era threshold for "fear you're becoming too advanced" based on the recipient's era
AdvCiv	*K-Mod*

When a civ receives a tech in trade, all third parties who know the recipient increase their received-tech memory about the recipient (as in BtS), except when (as in K-Mod) the recipient had already researched 2/3 of the tech, or when the era number of the tech is at least 2 lower than the era number of the recipient. E.g. when a civ is in the Medieval era and receives Archery, no tech-received memory is increased. The recipient's tech-received memory (for "You've shared your discoveries") is also not increased when one of the exceptions above applies.	Tech-received memory leads to "We fear you are becoming too advanced" (as in BtS). K-Mod adds the two exceptions. In the second one, game era (averaged over all civs) is decisive, not the recipient's era.

Rationale	K-Mod comment in `CvDeal::startTrade`: "This is to prevent the AI from being crippled by human players selling them lots of tech scraps." Good reason, but this also affects human players receiving tech. Humans need to be able to tell whether a trade is going to count, and the game era is unknown early on, and tedious to determine later on. I've extended the exceptions to "shared discoveries" because that's also exploitable in BtS.

Insignificant tech trades (2/3 progress or 2 eras behind; see above) are unaffected by the no-brokering game option. Regardless of the brokering option, such techs can be passed along on the same turn that they've been received. The AI does not reject any tech trades on account of the no-brokering option. Insignificant tech can also be traded immediately when it's received through some mechanism other than trade, e.g. the Internet or after the creation of a colonial vassal.	Techs can't be traded on the same turn on which they've been received in trade. When the AI has already 50% or more progress toward a tech, it refuses to accept that tech in trade when "No Tech Brokering" is enabled: "We would have nothing to gain." The tech brokering restrictions apply to all techs that a civ doesn't discover through its own research or Great People.

Rationale	Removing the 1-turn delay made it much easier to implement the change to "no brokering". Also makes sense to treat techs that are traded when 2/3 complete the same way as techs discovered entirely independently.
See also	Tech received through the Internet is actually not subject to any trade restrictions in BtS; kekm.31 removes that inconsistency.

550f	Tech purchases
See also	036 also affects the amount of cash that the AI is willing to trade.
AdvCiv	*BtS*

The target amount of gold that the AI tries to keep in its treasury can be increased beyond the BtS value depending on the potential for tech trades.	Based on civ size, leader flavor and units needing upgrades.
When the an AI civ checks for possible tech trades with another AI civ, if it doesn't find a tech-for-tech trade, then the first AI civ tries buying the tech that it is currently researching for gold (if the second AI civ knows that tech).	AI-AI tech trades are always tech-for-tech. One side can offer gold in addition or multiple techs, but just gold for tech isn't possible.
Rationale	Should help against AI civs falling far behind in tech. On the flip side, it means extra gold for the tech leader. That said, the AI doesn't pay very well for tech, especially when it has fallen behind (see 550a, 551).
The portion of its treasury that the AI is willing to trade to a human player is tied to the research progress of the AI. The portion is maximal when the AI has researched 75% of a tech and minimal at 25%.	Based on civ size, finances, gold previously traded (all still the case in AdvCiv).
Rationale	Should make it more likely that a somewhat fair gold-for-tech trade is possible when the BUG gold alert triggers.

550g	AI tech trade value adjusted to tech discovery value
See also	CFC post by me
AdvCiv	*BtS/ K-Mod*
The adjustment is between +27.5% trade value (AI pays extra for an especially useful tech) and -42% trade value (AI pays less if it has no real use for a tech). Based on intermediate results cached in `AI_bestTech`. For better or worse, those results are biased toward cheap techs, so the AI will be more inclined to pay extra for cheap techs and to pay less for expensive techs. Also, the cache only gets updated when the AI chooses new research, so the information can be outdated.	`AI_techTradeValue` and `AI_techValue` are completely separate functions. The former is mainly based on research cost, the latter performs a complex evaluation of the effects of a tech. Tech path, i.e. the utility of becoming able to research more advanced techs, are only evaluated by `AI_bestTech`, which returns a single best tech that the AI wants to research most.
Rationale	This is intended to be a moderate adjustment. It's smart to pay for tech based on its intrinsic value, but I don't want to make weak techs like Divine Right even more unattractive to research.
	I've dialed this up in v1.02 (the adjustment was previously between +25% and -33%) in order to weaken early-game tech trading a bit, specifically going for Aesthetics right after Writing and bubling Compass on the highest difficulty levels, and bee-lining to Alphabet medium difficulty levels.
See also	CFC post suggesting that religion techs make good trade bait because the AI won't be interested in researching them once the religion has been founded.

550h	AI research decision puts greater emphasis of possible tech trades
Credits	Suggested by Lanstro (CFC post)
AdvCiv	*K-Mod*

Made the AI more likely to take into account possible tech trades when choosing a tech to research. Tech trades now matter, typically, for about every third research decision. The probability is still personality-based.	When the AI chooses a tech to research, it decides randomly whether or not to take into account possible tech trades. The probability is equal to the contact probability for AI tech trades, which is between 5% (Tokugawa) and 100% (Mansa Musa); 20% for about 20 of the AI leaders; i.e. typically, tech trades are taken into account for every fifth research decision.
Also increased the utility value counted per possible tech trade.	K-Mod 1.46 had already increased the utility value counted per possible tech trade.
Rationale	To make it a bit harder for human players to take advantage of unpopular techs like Aesthetics. Mustn't take this change too far though 8see rationale under 550g).

551	AI trade value of tech reduced	
AdvCiv		*BtS*
When the AI trades for tech, the research cost of that tech is multiplied by 1.25 as part of the trade value computation.		The multiplier is 1.5. For comparison, traded gold gets multiplied by 2.
Depending on how widely a tech is known, its trade value is multiplied by a factor between 1 and 4/3.		The multiplier is between 1 and 1.5.
Rationale	Could argue that research and gold should have the same modifier because gold and beakers are essentially interchangeable. But gold should be more valued in trade because the other side loses the gold whereas imparting a tech costs nothing. By reducing the trade value of tech, I'm making it harder to conduct diplomacy (sponsored war, brokered peace, fair trade bonuses) through tech, and easier to buy tech with gold. I'm not setting the multiplier even lower because change 550a already makes it easier to buy tech (for civs that have fallen behind).	
Tbd.	The underlying issue is that it doesn't cost anything to share tech.	

552	AI trade modifiers (XML) revised	
AdvCiv		*BtS*
AI trade modifier reduced from 10 to 0: Archery, Horseback Riding, Guilds, Military Tradition, Ecology, Electricity, Advanced Flight, Composites, Stealth, Genetics, Fiber Optics, Satellites, Robotics Increased from 0 to 10: Civil Service, Nationalism, Chemistry, Military Science		In trades, the AI treats some technologies as having a 10% higher value than their beaker cost would suggest, namely those listed in the left column and (no change): Feudalism, Flight, Machinery, Gunpowder, Rifling, Steel, Assembly Line, Railroad, Artillery, Industrialism, Rocketry, Fission, Fusion.

Rationale	The BtS weights seem to be aimed at military techs, especially nuclear war, and Space victory. I think this should be (and is at least in part) handled by the DLL; don't want an AI civ that isn't aiming at a Space victory to pay extra for e.g. Genetics.
	I'm keeping the modifiers for military tech and I'm adding a few more, but not for pre-Medieval tech as pre-Medieval AI warfare is fairly rare. There is also `CvTeamAI::AI_getTechMonopolyValue`, which makes the AI not "want to start trading away this technology just yet," but if the AI still agrees to the trade, I want there to be an extra high penalty if it's a trade with someone's worst enemy.
Tbd.	+10% trade value is not going to affect the enemy trade penalty much. Perhaps 130p should give the AI trade modifier some extra weight when recording trade values (`peacetimeTradeValue`).
Config	`CIV4TechInfos.xml`

553	"Shared discoveries" relations modifier accumulates faster, decays faster.	
AdvCiv		BtS
The relations modifier from "shared discoveries" decays by one technology every 30 turns (on average). Each imparted tech adds between 0.08 and 0.2 to the relations modifier.		The memory count decays by one every 100 turns. Each tech adds a fraction between 0.05 (e.g. Alexander) and 0.2 (only Mansa Musa and Peter) to the relations modifier. (The sum of those fractions is rounded down.)
See also	Fractional relations modifiers are rounded to the nearest integer by 130j. This makes it much easier to reach +1 from "shared discoveries".	
Rationale	To balance out change 130j, and in the spirit of making diplomacy (a bit) more dynamic.	
Config	The respective `MemoryAttitude` values are modified through the DLL. (To avoid having to change the XML values of every individual leader.)	
Tbd.	Should change the `MemoryAttitude` values in `Civ4LeaderHeadInfos.xml` eventually, i.e. when I feel confident in my adjustments.	

groundbr	XML options for a research rate penalty when breaking new ground (disabled by default)
Credits	Based on the getSpreadResearchModifier function in "Dawn of Civilization"
Rationale	As a rubberband. I don't really think AdvCiv needs this; mainly for mod-mods.
See also	Discussion in the AdvCiv thread, on and off over several pages, starting with this suggestion by Cruiser76.
Config	See `TechDiffusion_GlobalDefines.xml`. The comments there also describe how the penalty is computed.
Tbd.	I don't think the AI takes research penalties into account when choosing its research.

570	Changes to expenses (tagged with "advc.exp") in the code	
advc.exp.1	Cap on number-of-cities maintenance	
AdvCiv		BtS/K-Mod

The city count in the formula for number-of-cities maintenance counts vassal cities half (as in K-Mod) and is capped at a map-size-based upper bound between 22 (Tiny) to 25 (Huge).	BtS counts vassal cities fully but caps the total at 4 to 8 cities (depending on the difficulty level) – i.e. vassal cities often don't matter at all. K-Mod halves the vassal city count and entirely removes the upper bound on the total city count in order to, as the changelog says, "*weaken the power of huge civs*."
Credits	Elkad made me aware that, without a cap, city maintenance makes Domination victories on Huge maps too difficult and incentivizes city razing. CFC post More recent CFC post by me (penultimate spoiler box) CFC post by drewisfat that calls the K-Mod change (well, he seems to attribute it to me) "insane."
Config	`MAX_CITY_COUNT_FOR_MAINTENANCE` in `GlobalDefines_advc.xml`.
Rationale	If I just restore the BtS cap, it'll (again) be far too easy, for large civs and in general, to max out the research slider, but the K-Mod change is too extreme to leave it alone. A cap around 40 would be enough to ensure that (decent) conquered cities can pay for themselves, but still leads to number-of-cities maintenance costs that are ca. 5 times higher than in BtS when aiming at Domination on a Huge map. A slightly better formula (for game balance) would take the city count to a power slightly below 1, e.g. 0.9, instead of capping the city count. Or some other soft-cap mechanism. See the second half of this CFC post of mine. However, opinions seem to differ on whether costs should simply grow quadratically as in K-Mod; so players may want to customize the formula, and a complicated formula would make that more difficult.
See also	140 adjusts number-of-cities maintenance to the crowdedness of the map.
Tbd.	Number-of-cities maintenance isn't the best tool for increasing the expenses of large civs. I surmise that it was intended only for pacing early-game expansion. I'd like to get rid of number-of-cities maintenance entirely; it's a clunky and slightly confusing concept. I hope that a fixed cost per city, 2 gold perhaps, or a bit less, say 5/3, would work well enough. Part of that cost should be paid by the capital in order to encourage a Courthouse there, or more generally, government centers should pay a maintenance cost for the number of cities in their vicinity (Voronoi cell). To increase expenses, civic upkeep should be increased, especially for civics in the Medium and High upkeep class. Ideally in a way that takes into account how spread out the population is; that could avoid punishing small cities that are mostly intended to claim resources. On that note, population should have a greater impact on distance maintenance and something needs to be done with colony maintenance, which discourages colonies too much. The Organized trait may have to be tweaked if civic upkeep is increased substantially. The trade route commerce fomula also needs work. There's generally too much trade route commerce in the midgame and late game, and coastal cities aren't rewarded enough. (Related CFC post – see item 4.) Inflation should be replaced with even higher costs for city population and civics and increased maintenance costs for units starting sometime in Renaissance (when units begin representing increasingly large numbers of soldiers. Also, air missions should have a small gold cost. (I've more detailed notes about all that, but no definitive proposal.)

advc.exp.2	Reduce effect of unique buildings on city maintenance	
AdvCiv		*BtS*
Ikhanda reduces city maintenance by 15% and Rathaus by 70%.		20%, 75% respectively.
Credits	Suggested [here](#) (last quote box) by CFC user AllTheLand.	
See also	[CFC thread](#) rating Ikhanda as one of the best unique buildings. [CFC thread](#) with praises (from reputable players) for Rathaus. exp1 (see above) reintroduces a cap on city maintenance – but still a much higher one than in BtS. [908b](#) weakens Terrace; a precedent for weakening powerful unique buildings.	
Rationale	These buildings aren't really problematic, but, compared with other unique buildings, they're already quite good in BtS, and the K-Mod increase to city maintenance makes them significantly more effective. Moreover, AdvCiv makes various changes that slow the pace of games on high difficulty levels and make it more difficult to decide games early on; those changes also make midgame buildings more useful. When the mod creates a problem, it's best to address it in some way, even if it's only a minor problem. Players tend to have limited tolerance for mods making any part of the game worse. The Ikhanda already had an unusual modifier (I can't think of any other 20% modifier in the whole game), so giving an unusual modifier also to Rathaus doesn't worry me too much.	

650	Changes to nuclear warfare	
Tbd.	AI still doesn't care to disperse its stacks.	
See also	[130q](#), [130h](#) about diplo effects of nukes [143b](#): changes concerning vassals and nukes [031](#) changes the AI evaluation of resources in a way that makes Uranium more attractive (once nukes can be built). [kekm.7](#): Neutral units not hurt by nukes [kekm.16](#): AI more willing to construct Bomb Shelters [kekm.20](#) increases the chance of civilian units getting killed by nukes. In particular makes Bomb Shelters less effective at protecting nuke units from enemy nukes. [906](#) moves missile-carrying submarines to Rocketry and lets Stealth Destroyer carry missiles. [045](#) hides city buildings from rivals. That means, `CvUnitAI::AI_nukeValue` shouldn't be based on non-visible buildings. See *Tbd*. under 045. [002m](#) cuts nuke animations short.	
AdvCiv		*BtS/ K-Mod*

	AdvCiv	BtS
	Damage from nukes to combat units is 9 plus the roll of a single 140-sided die, i.e. a number from the interval [10,149] is chosen uniformly at random for each affected unit. (All units have 100 hitpoints.) This means that, the per-unit survival chance is 9/14 (64.3%). The chance of surviving two nukes in a row is about 17% (if I'm doing my math right). The AI evaluation of potential target stacks takes into account the survival probability and the prospects of a follow-up attack with conventional units.	Damage is 28 plus the sum of two 50-sided dice, resulting in a survival chance of ca. 85% for a single nuke and a very small (somewhat difficult to calculate) chance of surviving two nukes in a row – provided that there are no Bomb Shelters. The AI does not try to predict how many units will be destroyed by a nuke. It does take into account remaining hitpoints, but, if I read the code correctly, in a way that makes the AI *less* interested in hitting injured stacks.
Rationale	The BtS formula makes it optimal most of the time to drop exactly two nukes on a large enemy stack, and this is unintuitive and uninteresting. A single nuke is only a good investment against the largest of stacks (or when able to follow up with a conventional attack), whereas two nukes eliminate pretty much all enemies. In particular, a reasonably strong AI city garrison can be wiped out reliably with two nukes, giving rise to a first-strike strategy that captures most coastal cities of an AI civ on the same turn as declaring war. While the AI is now somewhat wise to the double-nuke tactic, it still doesn't plan on dropping multiple nukes in a row, i.e. it should be able to figure out when it's a good play to nuke an injured stack, but it's not going to injure a stack through a first nuke with the intention of nuking it a second time afterwards. So nerfing this cheesy tactic also makes the AI more competitive.	
Config	`NUKE_UNIT_DAMAGE_`... parameters in `GlobalDefines_advc.xml`. The AI should be able to handle any nonnegative values.	
Tbd.	The AI still doesn't disperse its units in order to avoid nuke attacks. I'd like to further increase the probability of surviving multiple nukes. One could use a negative base damage value to this end, but would give tiles with one or a few units a chance of sustaining no damage to units at all, which seems strange. Units not getting damaged, in my mind, does not mean that the nuke wasn't well aimed (or the target information inaccurate), but that the units stacked in the tile are too far dispersed to hit them all. Such a model really only make sense when a nuke affects a single tile – and I think that would be better for game balance too. In the interest of reusing exiting AI code, I'd then try to use a formula that uses a base damage value of 100 when only a single unit is affected (guaranteeing the unit's destruction) and that adds a (single) die roll when there are multiple affected units, reducing the base value (to 0 and below) while increasing the maximal value of the die as the number of units is increased.	
See also	Deity K-Mod game won by double nuking AI city garrisons: CFC thread	
AdvCiv		*BtS*
The SDI has an interception chance of 60% (i.e. 30% for Tactical Nukes) and costs 1500 production (on normal settings), sped up by Aluminum as in BtS.	75% chance (i.e. 37.5% for Tactical Nukes), costs 1000 production.	

Rationale	Too much of a no-brainer once nukes become a threat and Aluminum is available, and makes ICBMs too much worse than Tactical Nukes overall. The AI isn't aware either that relying on ICBMs is a bad idea. Also, the high interception chance can easily lead to five or more ICBMs intercepted in a row. Which is how independent random trials work, but I think players expect that they'll be able to "force" a couple of ICBMs past it, and, at 75% odds, this goes very badly too often – I feel. I don't want to change the SDI much, however, because it'll always be unrealistic – if I make players relearn some element of play, I'd like it to be something sensible. 60% still means that interception is more likely to succeed than to fail.
Config	`Civ4ProjectInfos.xml`

The nuke target value is reduced based on the enemy interception probability (including the evasion chance). The interception probability has less impact if all current and plausible future enemies have a high interception probability. When hovering in Nuke Mode, help text shows the interception probability (if greater than zero).	Nuke interception is not taken into account when deciding whether and where to attack with nukes, and the UI only shows the evasion chance (in the unit help text of Tactical Nuke).

Rationale	Hover text: Easy to forget about the SDI, and it's not totally obvious how interception chance and evasion chance interact.
See also	004c shows the interception chance against air missions.
Tbd.	The cursor color in Nuke Mode leaks information about tile ownership in the fog of war. What to do? Require active visibility on all affected tiles? This would also eliminate the implausible recon visibility while the nuke explodes. It's also not plausible that a nuke fired without the latest recon info would be just as damaging, especially to units. Or introduce a `canNukeAt` check that works based on revealed ownership and allow for accidental declaration of war upon execution of the nuke mission? Ultimately, ICBM should move to Satellites, and then the Satellites ability could also lift the fog of war when it comes to tile ownership. Putting this effect at Rocketry (the current ICBM tech) would be less elegant.

An announcement summarizes the damage inflicted by nuke explosions: Destroyed and damaged units, destroyed buildings (only if visible), lost population, destroyed improvements and features.	Only announces that a nuke has exploded and where. The nuke grants visibility for its owner, but only while the nuke animation is playing, so the nuke owner may be able to glimpse some of the damage. Fallout shows where improvements and features have been cleared, but this doesn't tell players *which* improvements and features were destroyed.

See also	045 hides the 3D models of buildings in rival cities.
Rationale	For a player who knows the rules for nuke explosions, the destroyed buildings would seem like the most relevant information not provided by BtS. That is, only for the owner of an affected city. However, the nuke rules aren't necessarily that widely known, specifically how units get damaged. The AdvCiv changes to the damage distribution make it all the more important to be more transparent about the effects of nukes.

Tbd.	The announcement might be too detailed. Perhaps improvement and feature destruction shouldn't be included (already isn't included for third parties) and perhaps destroyed and damaged units should never be listed (even if the number is small), only counted. Even for the city owner, the details could be annoying – it's not so rare that a bunch of cities get nuked in a row.
	Would be nice to announce destroyed nuke units separately as they're especially expensive.

Academy, Military Academy and Scotland Yard are immune to nuke damage.	Great Wonders (incl. Shrines and corp. HQs), Bunkers and Bomb Shelter are immunte.
Rationale	Seems a bit too punishing to remove GP buildings, and consistent with Shrines and settled GP not to remove them. Ironworks getting destroyed is also brutal, but can't plausibly be protected either.
	Regular bunkers cannot plausibly withstand a nuclear explosion, but, given the ability of Bunker, it may well represent air raid shelters. While those aren't a sufficient protection against a nuke, it's plausible enough that they wouldn't be utterly destroyed.
Credits	Idea from [Dawn of Civilization](#)
Tbd.	Would prefer to let nukes destroy obsolete wonders and settled GP. Will need to announce those effects though. Obsolete wonders also shouldn't be immune to capture-city damage.
	Bomb Shelter should be renamed to "Fallout Shelter."

AI uses air recon along its coast. Based on BBAI code for (human) auto-recon.	AI seems to use air recon only on rival cities.
Rationale	Could reveal enemy submarines. Not enough to properly defend against Tactical Nukes, but the BtS AI doesn't even seem to try.
See also	A bugfix in `CvUnitAI::AI_exploreAirPlotValue` is tagged with "[advc.001](#)".

Treat preparations for "total" war the same as ongoing or imminent "total" war in `CvPlayerAI::AI_nukeWeight`.	
AdvCiv	*K-Mod*
AI willing to build the Manhattan Project unless it looks like it will win the game anyway,	AI considers the Manhattan Project only when in the Dagger or Crush strategy, or when close to a Conquest victory.
Rationale	The AI shouldn't take risks with nukes when it's winning anyway (Conquest, or crushing its current war opponents).
AI less worried (divisor 3) about attitude of AI civs that aren't (prospective) war enemies when deciding whether to build the Manhattan Project. Not worried at all about unlocking nukes for rivals that are about to win the game (victory stage 4). Utility value for gaining access to nukes takes into account the number of civs in the game. Somewhat more sophisticated logic for figuring out who is on track to winning. May go for the Manhattan Project when feeling existentially threatened (but not when already losing a war – too late by then). As for war plans, only a "total" war in preparation can encourage the Manhattan Project.	The K-Mod AI checks the attitude of all rival civs; divisor is 2. Utility for nuke access is a constant.

Any war ongoing or planned can encourage the Manhattan project. |

Rationale	A K-Mod comment says that it should be up to the human player to decide whether he/she wants nukes in the game. I don't agree with that, but I mustn't make the AI so willing that some AI builds Manhattan most of the time; or at least not until the game is about to end. Generally, I'd like the AI to unlock nukes when a disliked civ is about to win the game.
Tweaks to AI nuke weight: Incentivize getting a first nuke when having none. Take into account personal espionage weight, war probability at Cautious attitude and favorite civic (Environmentalism) too.	The overall nuke weight determines how willing an AI civ is to produce nuke units. As for personality, only the peace weight value matters.
Rationale	All the Cold War leaders have high espionage weights.
Added a high-priority rule to the City AI for producing at least a small nuclear arsenal as soon as nukes become available.	
Rationale	It takes the BtS/ K-Mod AI too long to produce its first nukes, which make a big difference for deterrence both in gameplay and story terms.
Revised the calculation of the AI weight for nuking civilian targets: Enemy war successes increase the weight only if the war is going very, very badly. Decreased weight when a war should be easy to win judging from the power ratio. Leader's personal raze-city weight taken into account. Reduced impact of attitude. (Estimated) number of enemy nuke taken into account (potential for retaliation). The number of enemy nukes also affects the threshold that potential target tiles need to pass in order to be nuked by the AI.	This weight factors into the evaluation of target tiles for nukes, specifically whether to focus on hitting military units or on damaging the enemy's economy. If no tile is deemed sufficiently valuable, then no nuke is launched. The weight is computed from the war success rating, attitude and memory of enemy nuke attacks. The Conquest victory strategy also increases the weight.
Rationale	See comments in the code (`CvPlayerAI::AI_nuke...DestructionWeight`).
Tbd.	My estimate of enemy nukes uses the exact number obscured by random noise. Perhaps a similar statistic could be made available to human players so that this wouldn't be a cheat. For example, the number of nukes per rival could be shown on the Military Advisor under the minimap (and in the scoreboard tooltip of the power ratio) in intervals "1 to 4", "5 to 9", "10 to 16", "17 to 24" etc. (i.e. the more nukes a rival has, the less precise becomes the information). Alternatively, the intel ratio planned for change 082 could also apply to nukes (would still be a cheat).
Substantially reduced the impact of buildings on the AI choice whether and where to attack with nukes – by taking into account the chance of buildings surviving a nuke explosion intact.	City buildings seem to dominate the AI calculations, at least when no large garrison is present.
Tbd.	May have to dial up other factors to avoid making the AI too shy about using nukes. Let's see about that ...
AI doesn't disband nukes when in financial trouble unless the only other remaining units are cargo units.	As far as I can tell, nukes are disbanded before most other units. BtS tried to save expensive units, which is fine in the case of nukes, but K-Mod focuses on XP, which is generally better, but fails for nukes.

651	Changes to the Fallout Feature
Tbd.	Would like to rename this to "Exclusion Zone", remove the Scrub action and place only a single one after a meltdown and none after a nuke.

AdvCiv		*BtS*
Can't found cities on Fallout. Nor on Ice.		Oasis is the only feature that prevents cities. Coast and Ocean terrain also prevent cities.
Rationale	I think it's a bit more intuitive this way. (Ice only matters for the Civilopedia text; if water is explicitly forbidden, it seems cleaner to also forbid Ice.)	

652	Rule changes to Meltdowns
See also	106 reports meltdowns more widely. kekm.5 disables the meltdown chance when there is a clean alternative power source.

AdvCiv		*BtS*
Adjust the meltdown probability to game speed.		Not affected by game speed.
Rationale	For balance – to match the higher re-construction cost and amortization time; for immersion – nuclear disasters shouldn't be more common on slower game speed.	
Show the meltdown probability in help text. Omit the ability in help text for the city building list when the Nuclear Plant isn't providing the city with power.		Help text always shows a "small" chance of a nuclear meltdown.
See also	Based on kekm.5.	
Increased the meltdown chance to 0.7 permille.		0.5 permille.
Rationale	I did some math (see below), but ultimately went by intuition. The BtS probability seems a bit too small now that meltdowns are less disastrous (K-Mod change) and can't happen when the Nuclear Plant is superseded by another power source (Kek-Mod). On the other hand, games tend to last longer with AdvCiv than with K-Mod ...	

On Normal speed, there are 125 turns left in 1955 (turn 375), but games hardly ever last that long. A more typical end date for a peaceful AdvCiv game is 2005 (turn 425), i.e. just $T=50$ turns of nuclear power. If p is the meltdown probability and one assumes (for simplicity) a constant number N of nuclear plants throughout the T turns, then the total number M of meltdowns is distributed binomially with success probability p and $N*T$ trials and thus an expected value of $EM = p*N*T$. Aim at $EM=1$ (Chernobyl), then N as a function of p becomes $N(p) = 1/(50*p)$. Table:

```
p       N
1/2500  50
1/2000  40  (BtS)
1/1500  30
1/1250  25
1/1000  20
1/750   15
1/500   10
```

A Standard size map only has about 50 cities in the late game. Considering that some civs will reach Fission extra late or never and that one (big) civ may build Three Gorges, N=10 might be a realistic assumption for an all-AI game. In this human game, there were 33 Nuclear Plans at game end, which seems unusually high, – but half of those are out of order due to another power source.

However, if the meltdown probability is greatly increased, then long games on larger

	maps could end up with a high total number of meltdowns, and already an expected M=1 on Standard size can easily result in multiple meltdowns. You'd think that, after the first one or two meltdowns, people would so something to secure those plants or else abandon them ...
	A map size adjustment could be justified by arguing that a single Nuclear Plant represents several plants on smaller maps, but it would still make Nuclear Plants less useful on smaller maps than on larger maps, which is counterintuitive.
	Looking at it from the point of view of a single plant, the chance d of dying to a meltdown over a 50-turn span is $1-(1-p)^{50}$, i.e. ``` p d 1/2500 1.9% 1/2000 2.5% (BtS) 1/1500 3.3% 1/1250 3.9% 1/1000 4.9% 1/750 6.5% 1/500 9.5% ```
	At some point, it becomes implausible that someone would construct such a time bomb.
See also	In the 4th bullet from the end, this post mentions an experiment to assess the proper meltdown chance. (Better to stick to the math though.)

700	Rise & Fall game option; see chapter Rise & Fall. Id 700: the fundamentals. Other ids only for code outside the R&F classes. Except for those highlighted in blue, all changes in the following are only relevant if R&F is enabled.
Rationale	For the term "intermediate period": Ancient Egyptian periodization (Wikipedia).
Config	The number of chapters, interlude length and scoring delay are configurable through `GlobalDefines_advc.xml`.
Tbd.	Earth1000AD scenario not currently supported. During initialization, `CvPlayer::verifyAlive` kills all civs because no cities have been created yet. I don't see how the R&F code causes this, but apparently, it does. Also, the unequal AI handicaps aren't currently supported by R&F. Would have to store the AI handicap (in `RFChapter` and in savegame) before human takeover and restore it when AI control resumes.
	Should perhaps factor game settings other than difficulty into the Rise score, in particular the number and length of chapters. For now, my assumption is that a higher number of chapters implies a lower number of turns per chapter, which makes it harder to maximize the chapter score, and that this about cancels out.
Credits	Inspired by board games like Peloponnesian War and History of the World (both from 1991) and Kael's Assimilation mod (which is included in Fall From Heaven II and RoM).
See also	Since R&F scoring is partially based on victory stages, it somewhat hinges on change 115, which revises the computation of those stages.
	044 autosaves at the start of the active player's turn. R&F does so too, but the exact timing is a bit different.
701	"Require complete kills" option replaced by the R&F option
AdvCiv	*BtS*

When the last city of a civ is captured, all its remaining units are removed, and the civ is eliminated; there is no option for changing this behavior.	When "Require Complete Kills" is checked, units remain in the game when the final city is captured. A civ is only eliminated when all of its units are killed and all cities captured.
	The option was added by patch 1.61, apparently mainly for mods, specifically Jon Shafer's "Battle for East Asia".
Rationale	I had already disabled the complete-kills option (for the reasons stated below) before adding the R&F option. Removed it entirely now because it's easier to replace a game option than to add a new one (while maintaining savegame compatibility).
	Too much of a hassle to keep the complete-kills option in mind for all the AI code. I doubt that the WL, BtS, BBAI and K-Mod code can really handle it either, especially in team games when some team members still have cities and others don't. Not sure if this option was ever intended to be more than a gimmick. You even need to kill every enemy Spy (or since BtS: hope that the Spy kills itself somehow).
	Now, in principle, the AI still needs to be able to handle civs without cities because a human player could decide to never found a city. Through Advanced or later-era start, the player could even have a sizable army, not to mention scenarios. I'm trying to have the game at least not crash in such cases, but it's not really supported. (Barbarians don't necessarily have cities either, but get special treatment in most of the AI code; this works.)
Tbd.	Instead of just removing the units, remove some, turn some into Barbarians and turn some over to any teammates or (war) allies that the dead civ might have had before its demise. This is more plausible and doesn't create issues for the AI.
Rise & Fall option not visible on the Staging Room screen.	
Rationale	Not multiplayer-compatible.

702	Error handling if game settings are incompatible with R&F
Not compatible: multiplayer, team games (including Permanent Alliances), One-City Challenge, SPaH.	

703	"Score" tab on the Victory advisor screen
Showing scores of past chapters and the start dates of chapters still to be played. When R&F is enabled, the tabs "Resolutions" and "Members" are merged into one in order to make room for another tab.	

704	Changes to Dawn of Man (DoM) screen
Tbd.	Adjust the layout so that the traits can be spelled out. Could be adopted from History Rewritten, but I've made my own mockup, which I like better.
Show the screen at the start of each chapter.	
Tbd.	In the later chapters, the DoM can appear with a few seconds delay. Seems like the EXE is too busy with something else. Perhaps this wouldn't happen if I'd launch the popup earlier.
See also	004j: DoM screen shown after regenerating map 004x shows a choose-civics popup after the DoM screen when starting in a later era.
AdvCiv	*BtS*
Heading of the DoM screen says "The Dawn of Civilization".	"Dawn of Man"
Rationale	Perhaps they were thinking of Kubrick's "2001", which shows a subtitle "The Dawn of Man"; but that is followed by a scene with apes ...

705	Rules and AI changes to prevent self-collusion (helping a civ from the previous chapter that hasn't been scored yet, or one about to be taken over)
In addition to the AI behavior described in the R&F chapter:	
The AI is a bit more lenient in pre-Currency tech trades – accepts if it receives only 90% of what the player receives.	
If UWAI is enabled, the AI refuses to talk when war utility is 20 or greater.	
Rationale	During a chapter, the player should (ideally) only have to consider the interests of his/her current civ. The AI changes described above (and in the R&F chapter) combined with intermediate periods can't completely prevent self-collusion but make it much less relevant. Restrictions don't apply to civs that the player is probably not going to want to play in the next chapter.. Trade acceptance: With R&F, it can be extra annoying when the AI refuses to trade two techs of (almost) equal value, but won't accept a 2 for 1 either because this would already favor the AI too much. War utility threshold: Don't want players to bypass trade restrictions through reparations. Would be nice to give the player more feedback, e.g. a grayed-out "gift" button, or the AI responding "you're being too generous" when an offer is too good, but these things seem almost impossible to implement. I did manage to get the AI to say "no thanks" to gifts.

Tbd.	Perhaps the AI should always show some leniency in pre-Currency trades, i.e. regardless of the R&F option.
	Perhaps reduce or remove the penalty for playing a civ multiple times.
	Should the AI be willing to talk when all potential peace deals are too favorable for the AI? E.g. when the AI wants the equivalent of 1000 gold and the player doesn't have that much gold but does have some 5000-beaker technology. Currently, the AI is willing to talk then, and that's a bit confusing. Though refusing to talk would also be confusing I think – "How is that big technology not enough?"
	The AI currently accepts certain resource trades that have a benefit for the AI side but no benefit for the human side. For example, if the AI has 2 Rice and needs more health, and the human has 0 Rice, the AI will accept a human non-surplus health resource in exchange for the Rice. Such a trade could make sense for the human player because of buildings like Granary, but, normally, it'll only help the AI. I have some notes archived (offline) about how to fix this, but I'm not sure if it comes up often enough to bother.

706	Changes related to AI Auto Play and switching the human-controlled civ
When the player "Retires", the current chapter completes on AI Auto Play. The intermediate periods in between chapters aren't actually AI Auto Play; there simply isn't a human-controlled civ at all.	
I hide some UI interface elements during intermediate periods and suppress advisor screens and popups. This has required changes in a bunch of different places.	
Rationale	The player may simply want to end a chapter because he/she doesn't feel like completing it. But retirement is mainly intended for situations when things are going *too* well. The civ could then become a huge obstacle in later chapters (or just win the game for the AI). Don't want the player to sabotage his/her own civ then.
Tbd.	After retiring, the player's civ keeps the game handicap, e.g. Prince, but, since the civ is treated as an AI, the AI modifiers from the game handicap also apply, e.g. 5% faster training of units on Prince. This is confusing, but difficult to change because the AI modifiers get applied in many different places. AI Auto Play without retirement applies the AI handicap (change 127), which is consistent, however, a retired civ should generally do rather badly because retiring is supposed to help when the player's civ gets dangerously far ahead. Perhaps set the handicap after retiring one higher than the game handicap (or always Monarch?) to somewhat cancel out the AI modifiers.
	Known issue: When taking control of a civ via Civ Changer (Alt+Z) for debugging purposes (in a R&F game, to be clear), all non-expired messages that that civ ever received are displayed on the main interface. Calling `CyInterface().clearEventMessages()` from `ChangePlayer.py` doesn't get rid of them.
See also	Hinges on 127, which allows MoreCiv4lerts to work when the player takes control of an AI civ. Also records start and end of AI Auto Play in replays, which makes it unnecessary for R&F to record early retirement in replays.
	The K-Mod code that 058 (re-)moved was messing up the interface messages sent to AI civs (which the human player may eventually read after switching civs).

707	Changes to the game end sequence
No game end upon defeat; Rise score shown als "final score" in HoF, scaled up ("normalized score") for the Dan Quayle screen.	
See also	043: Uses stricter thresholds for the titles on the Dan Quayle screen.

708	Player handicap increased above game handicap	
R&F		*BtS*
The game handicap is set as configured on the Custom Game screen, but the human player handicap is set one higher than that and the AI player handicap is set to Prince. For example, when Prince difficulty is set on the Custom Game screen, the inflation modifier for AI civs is 95% from the AI player handicap (Prince) times an 80% discount from the game handicap (Prince), while the inflation modifier for human civs is 100% from the human player handicap (Monarch). That means, the AI inflation modifier is 76% of the human inflation modifier.		The human player handicap and the game handicap are always both set to the difficulty level configured during game setup. The AI player handicap is always Noble. The game handicap is reponsible for AI freebies and discounts and Barbarian activity (but not combat modifiers and free wins against Barbarians), the player handicap for everything else. That is, a difficulty other than Noble is chosen, the uneven playing field between human and AI players is the result of them using different player handicaps combined with the AI (dis-)advantages set by the game handicap. For example, if the game handicap is Prince, the inflation modifier for AI civs is 90% from the AI player handicap (Noble) times an 80% discount from the game handicap (Prince), while the human inflation modifier is 95% from the human player handicap (Prince). That means, the AI inflation modifier is ca. 75.8% of the human inflation modifier.
Config	The player handicap adjustment can be disabled or increased through `RF_PLAYER_HANDICAP_ADJUSTMENT` in `GlobalDefines_advc.xml`.	
Rationale	Playing two or three difficulty levels below one's habits or abilities is fun in some ways – in particular being at a level playing field with the AI civs –, and is really one of the main points of the R&F option, but not much fun in other ways. Coming from, say, Monarch, the maintenance costs on Noble will (rightly) seem quite insignificant, which can trivialize some important strategic decisions. (Perceptions matter a lot for expenses I think: a player who is scared of city maintenance will weigh his or her decisions about the pace of expansion.) Free wins against the Barbarians also feel cheesy. The intention is not to make the game easier or harder in terms of competing with the AI civs (which is what R&F scoring cares about) – players know how challenging the AI is on Noble difficulty, and that's what they should get when they select Noble. Admittedly, the progression of difficulty-based modifiers is not entirely smooth, so increasing everyone's player handicap probably does make it either a bit easier or harder to beat the AI civs (I really don't know which it would be). It might be nice to use a higher handicap also for some aspects of the game handicap – essentially everything except the ongoing AI discounts –, but seems difficult to say and would make the handicap mechanics even more complex (and would be more work to implement).	
See also	251 increases some expenses for the medium difficulty levels and reduced the number of free wins against Barbarians. 127: The difference between game handicap and player handicap is also relevant for AI Auto Play. CFC post by a Deity player characterizing city maintenance on Monarch and below as	

	being of "*little to no impedance to your expansion*."
Decreased all research costs by 3% (through the DLL), in part offsetting the increased player handicap (which implies increased research costs).
Rationale	A human player with skills well above the player and game handicap always being in charge of one civ should lead to faster research overall than one would expect from e.g. a game on Noble difficulty (game handicap), but, then, the human player is, for much of the game, in charge of a civ that catches up, so it would seem that the proper cost should be somewhere between the game handicap and (increased) player handicap.
See also	910 is more generally concerned with having the pace of research match the historical time line (game date).

901 et sq.	Changed stats		
Config	mostly XML-based		
See also	310 (Great Wall, Great Lighthouse, Colossus) would also fit here. 160 lets Granary store less food. 650 makes changes to the SDI (and to the effects of nukes)		
Tbd.	See this draft of long-term balance and historicity changes.		
901	*AdvCiv*	*K-Mod*	*BtS*
Forest Preserve	at Biology; +1 commerce	at Scientific Method; +1 commerce; +1 commerce if riverside	at Scientific Method; +1 commerce if riverside

K-Mod 1.45 has added a 7th effect to Scientific Method, and enlarged all tech boxes to make room for a 7th icon. Not a good solution; surely no single tech needs to do 7 things.

Nature reserves originated in the early 19th century according to Wikipedia, which fits with Biology. I also don't see much of a connection between scientific method and forest preserves. Not a question of game balance really because Forest Preserves are rarely useful.

Removed the riverside commerce while I was at it. Simpler this way. Not much of a loss because Forests along rivers get chopped anyway (see also change 117). Consistent with Lumbermills, which also provide commerce regardless of river in AdvCiv.

AdvCiv		*BtS*
	Positive happiness and health effects from surrounding tiles (features and improvements) only benefit cities that can peacefully enter the borders of the tile owner.	Tile ownership never matters; a Forest Preserve benefits even cities of a war enemy if it is in their radius.
Rationale		Unexpected that other civs can benefit from Forest Preserve – but perhaps not so unexpected for Forest health, and I want the same rules for both. Based on Open Borders as a compromise and because it makes sense that all citizens who are able to visit a Forest or Forest Preserve are healthier or happier than any citizens are unable to go.
Negative effects apply regardless of ownership (as in BtS).
Rationale		Don't want players to work around bad health from Jungle by keeping their borders closed. The disease carriers are mostly insects; borders don't stop them.
When a tile with a happiness or health effect becomes accessible or inaccessible, the city's health and happiness aren't updated until end of turn.

Rationale	Too much work to make sure that updates happen whenever tile ownership, team membership, vassal agreements or Open Borders change.

Added an XML tag `HealthPercent` to CvImprovementInfo so that Forest Preserve can grant health instead of happiness. That health can be fractional and is added to health from features (rather than being rounded separately). With AI and UI support (both of which were horrible to write because of rounding – separately for good and bad health effects – and the possibility of muliple cities being affected). For now the tag remains unused, but some of the new AI code is used for evaluating the removal of Forests and the happiness ability of Forest Preserves. Negative effects are now also supported.

Rationale	Not urgent enough to change before AdvCiv version 1.0. By the time that Forest Preserve becomes available, extra health tends to be more valuable than extra happiness. Livability can be represented just as well through health as through happiness. (I'd prefer to use happiness mostly for effects with a fairly direct political dimensions, e.g. ludi at the Colosseum, luxury resources handed out as perks.)
See also	Some discussion about this on CFC (1 2 3) UI code based on 059. 055 adds an (unused) improvement ability that protects terrain features from Global Warming.
Config	It's an optional tag. See `Civ4TerrainSchema.xml` to find out the proper position in `Civ4ImprovementInfos.xml`.
Tbd.	Health is is currently overcounted; see comment in `CvCityAI::AI_healthHappyImprovementValue`. Make the Forest spread ability more predictable and more useful. Grant health instead of happiness. CFC post

902	AdvCiv	K-Mod	BtS
Watermill	+1 commerce and +1 production initially; +1 commerce with Electricity; 4 turns to build; +1 production with Replaceable Parts	+1 commerce and +1 production initially; +1 commerce with Electricity; 8 turns to build +1 production with Replaceable Parts	+1 production initially; +2 commerce with Electricity; 8 turns to build +1 production with Replaceable Parts

Watermills are decent with Replaceable Parts but rarely useful before that (whereas, historically, they were highly useful throughout the Middle Ages). The K-Mod change to Serfdom (+1 commerce from Farms) exacerbates this. One commerce at Machinery (also a K-Mod change) is the least that needs to be done.

Commerce from Electricity is apparently supposed to get the shift from waterwheels to turbines across (similar for Windmill). That's OK (electricity can be sold), but doesn't have to be so pronounced. Electricity boosting local production is also plausible.

Never saw why Watermills should be slow to build. Perhaps originally an attempt to make Serfdom attractive – well, that didn't work. Reducing build time is a way to make Watermill better early on without making it too good later on.

Tbd.: Still a bit unattractive. 6 turns to build but +2 production earlier?

Lumbermill	at Guilds +1 commerce initially; +1 production with Replaceable Parts; 5 turns to build +1 production with Railroad track; doesn't remove Forest	at Guilds +1 river commerce, +1 production initially; 8 turns to build +1 production with Railroad track; doesn't remove Forest	at Replaceable Parts +1 riverbank commerce, +1 production initially; 8 turns to build +1 production with Railroad track; doesn't remove Forest

The middle column is K-Mod 1.45; I'm keeping the earlier version (not shown above) and I'm reducing the build time in order to give Lumbermill a little extra push. Also, Replaceable Parts is just one tech away from Steam Power, which grants +50% Worker speed, but since Lumbermills are now at Guilds, that bonus is still out of reach, so they really take a long time to build in K-Mod.

I don't think players can be incentivized to not chop all Forests along rivers; not without making Lumbermills too strong in the late game. Therefore, I don't like the river commerce ability (not on Forest Preserve either).

Tbd.: Want to move both Lumbermill and the +50% chopping yield (currently at Mathematics) to Machinery eventually, and apply chopping yields only to buildings and ships (not land units).

903,904	reserved		

905a	*AdvCiv*	*K-Mod*	*BtS*
Galley	3 moves, 3 cargo, cost 45 Barbarians: Disorganized -10% strength, -1 move	2 moves, 3 cargo, cost 50 Barbarians: Disorganized -10% strength	2 moves, 2 cargo, cost 50 No penalties for Barbarian Galleys

Trireme	3 moves,
	3 strength,
	no bonus vs. Galley,
	upgrades to Caravel
	  and Privateer,
	cost 45
		2 moves,
		2 strength,
		+50% vs. Galley,
		upgrades to Caravel
		  and Frigate,
		cost 50	

Caravel	4 moves removed AI types "Reserve" and "Escort"	3 moves	3 moves
Carrack	4 moves, 2 cargo, removed AI type "Reserve"	3 moves, 3 cargo	3 moves, 2 cargo
Navigation2 promo	Only available for units that can enter ocean	Available for all sea units	

Should give Galleys a use in early warfare (fast deployment), and make them better explorers. If 3 moves aren't enough to make naval deployment useful in land wars, then 4 moves with Navigation1 should do the trick. Sea units moving twice as fast as infantrymen along roads is reasonable from a historic pov, but 5 moves might be a bit much, also gameplay-wise. Therefore, I'm restricting Navigation2. Makes sense that "navigation" benefits ocean-going vessels more.

4 moves for Caravel to keep it faster than the early ships. Now just as fast as Galleon, which is also more accurate historically. Carracks, caravels and galleons could travel at 4 to 5 knots, triremes at 2 to 3 (much faster during battle; see e.g. Olympias on Wikipedia).

3 strength for 45 production makes Trireme a viable warship (perhaps even the best) until Frigate, which should make Triremes more worthwhile overall and might help the AI, which tended to train too many Caravels previously. Should now train a mix of Triremes and Caravels, and use only Triremes for protecting resources and escorting Galleys. May still train too many ships, but at least Trireme is cheaper than Caravel. While I'm at it, the upgrade from Trireme to Privateer is a bit less painful than to Frigate because Privateer is cheaper than Frigate, and this should make AI privateering a bit more common.

Historically, trireme-based ships remained in use in the Baltic Sea until the 18th century, and their usage didn't decline until the 16th century. "[The Battle of] Lepanto [1571] marks the last major engagement in the Western world to be fought entirely or almost entirely between rowing vessels, the galleys and galeasses that were still the direct descendants of the ancient trireme warships." (Wikipedia)

Carrack: K-Mod had increased the cargo capacity along with the other transport ships, but I think this takes the unit unnecessarily close to Galleon, especially now that I've increased the speed (can't not increase Carrack speed if I increase Caravel speed). Back to 2 cargo spaces.

Tbd.: Require Optics or Astronomy for Navigation2; that's more elegant than the restriction to oceangoing ships. Galleys with 4 moves should be more common; remove the Flanking I requirement from Navigation I. But will then have to find another way to make Flanking I attractive, and Trading Post will have to change.

"Explore rival territory" of Carrack should perhaps not work with military units in cargo (unless those units somehow also have that ability).

See also: 124 assigns Galley the Explore AI type.
306 changes research costs to make Trireme a bit easier to access.
CFC post suggesting that 3-move Barbarian Galleys appear out of the blue too often.

905b	AdvCiv	K-Mod	BtS
Galleon (as in K-Mod)	4 moves 4 cargo 4 strength cost 80	4 moves 4 cargo 4 strength cost 80	4 moves 3 cargo 4 strength cost 80
Transport	5 moves 5 cargo 14 strength cost 100 requires Oil can't attack	5 moves 5 cargo 16 strength cost 125 requires Oil or Uranium	5 moves 4 cargo 16 strength cost 125 requires Oil or Uranium
East Indiaman	unique Frigate 5 moves 3 cargo 8 strength	unique Galleon 4 moves 5 cargo 6 strength	unique Galleon 4 moves 4 cargo 6 strength

	bombard rate 12
	cost 90
	upgrades to Destroyer
	  or Transport
	req. Astronomy,
	  Chemistry, Iron
		can explore rival territory
		cost 80
		upgrades to Transport
		
		req. Astronomy
			can explore rival territory
			cost 80
			upgrades to Transport
			
			req. Astronomy

Privateer	5 moves	4 moves	4 moves

Frigate	5 moves
	8 strength
	bombard rate 12
	cost 90
	upgrades to Destroyer
	req. Astronomy,
	  Chemistry, Iron
		4 moves
		8 strength
		bombard rate 8
		cost 90
		upgrades to Destroyer
		req. Astronomy,
		  Chemistry, Iron
			4 moves
			8 strength
			bombard rate 8
			cost 90
			upgrades to Destroyer
			req. Astronomy,
			  Chemistry, Iron

Ship of the Line	4 moves
	10 strength
	+25% vs. Frigate
	bombard rate 16
	cost 110
	req. Astronomy, Military
	  Science, Iron
		3 moves
		10 Strength
		+20% vs. Frigate
		bombard rate 12
		cost 120
		req. Astronomy, Military
		  Science, Iron or Copper
			3 moves
			8 strength
			+50% vs. Frigate
			bombard rate 12
			cost 120
			req. Astronomy, Military
			  Science, Iron

Credits: CFC user vedg pointed out here that there aren't enough incentives for upgrading Galleons and East Indiamen.

Rationale:

Transport: Having just 25% more cargo space than Galleon doesn't justify a more than 50% higher cost. Often not worth upgrading. To balance out the lower cost, I've removed the offensive abilities, which aren't entirely unimportant against rivals without Combustion. The unit can still "besiege" water tiles, i.e. prevent them from being worked. The lower strength should give pre-Combustion ships a fighting chance against besieging Transports, and shouldn't make much of a difference against post-Combustion units. While I was at it, I removed the Uranium requirement for historical accuracy; see this CFC discussion.

East Indiaman: Also no incentive to upgrade to Transport. On top of that, historically, East Indiamen appeared a century later than galleons, and were usually larger than frigates. Rather than a warlike Galleon, this is now a Frigate with cargo space. The ability to enter rival borders is messy on cargo ships because these ships (and their cargo) don't get bumped upon declaring war (see *Tbd.* under 905a). The Civilopedia states that "the East Indiaman was inevitably slower than its descendants: the Ship of the Line and the Frigate." This appears to be nonsense: East Indiamen were often full-rigged clippers that could sustain speeds of 10 knots, whereas ships of the line sailed at perhaps 5 knots.

Bombard rates: 8 is the same as Catapult, far too little for ships equipped with cannons. Did not increase Ironclad's bombard rate (12 as in BtS) because at least early ironclads weren't as heavily armed as ships of the line.

Speed increases: Even early frigates appear to have traveled at a speed of about 10 knots; much faster than galleons. According to Wikipedia, "a long hull-design, which relates directly to speed" was characteristic for frigates. While early privateers were galleons, the Privateer unit in the game is contemporary with Frigate and should therefore have the same speed. Ships of the line traveled at about 5 knots, which is much slower than frigates, but rather faster than galleons and caravels. Assuming some sort of logarithmic scale, speed 4 is justifiable. Speed 3 made it too difficult to use Ship of the Line offensively. See also this post of mine on CFC; thanks to Pepo for pointing out that Frigate needs to be faster.

Ship of the Line: I don't love the K-Mod change to strength; it makes Ship of the Line stronger, which is good, but does so partly at the expense of Ironclad, which is also too weak. 120 production seems pretty clearly overcosted. I've restored the BtS Iron requirement because it doesn't make sense from a historical point of view to require Iron for Frigate and not for Ship of the Line. The latter requires more metal parts (cannons, cannonballs, nails(?)), and thus has more need of a cheap metal. The K-Mod changelog says the goal was to give "ironless civs a decent naval ship if they go for the right tech." Fair enough – but not really important. I hope my other changes to Ship of the Line can give the unit more plausible (niche) uses.

Tbd.:

Bombard rates are still a bit too low. Ship of the Line and Ironclad need to be buffed further, and Frigate should upgrade to Ironclad. Want to allow Transport, Destroyer and Battleship with Coal, but with a movement speed penalty. Move Transport to Industrialism and Destroyer and Battleship to Artillery.

Could give Frigate and Ship of the Line +1 strength to make both more effective against Privateer and then restore Ship of the Line's 50% bonus against Frigate.

Frigate should be allowed to carry 1 "special" unit, or perhaps even a military unit, and should be able to enter rival borders (so long as no military units are carried). Caravel and Carrack should then upgrade exclusively to Frigate.

See also: 081 about AI changes to employ naval bombardment in land wars.

Config: `Civ4UnitInfos.xml`; note that East Indiaman is called `NETHERLANDS_OOSTINDIEVAARDER` in that file.

Ironclad	3 moves
	+25% Coast defense
	req. Steam Power
		3 moves
		req. Steel and
		  Steam Power
			2 moves
			req. Steel and
			  Steam Power

Rationale: To match the increased strength of Ship of the Line. The defense ability is a bit more flavorful than just +1 strength (and 13 would be an unusual strength value); ironclads had difficulty sinking each other and their battles tended to be stalemates. There is no generic defensive bonus in Civ 4 (though it would be easy enough to implement one), hence the terrain restriction. Note that cultural borders can allow an Ironclad to enter Ocean, so the restriction is not entirely irrelevant.

Removal of the Steel requirement gives Ironclad a longer window of utility. Also, the first ironclads, produced in 1862 for the US Civil War, were not clad in modern steel: the first steel mill in the US based on the Bessemer process was set up in 1865.

Tbd.: Stalemates aren't always fun; if it turns out that the ability plays badly, I could still just increase combat strength. It may also, despite the K-Mod change, still be too difficult to engage faster ships. Even early ironclads were only a few knots slower than frigates, so speed 4 would be justifiable; the restriction to Coast would still get across that ironclads aren't as mobile as frigates.

XML tag added for extra speed with access to a resource. Unused for now. "Access" means that either the unit's current tile or the owner's capital has access to the resource. The tactical side of the AI (`CvUnitAI`) is aware of the increased speed, but I haven't changed the strategic evaluation (`CvPlayerAI`) of resources and units.

Sevopedia articles about resources now have a "Units" box instead of "Allows" so that units can be listed in articles about resources. Buildings that require resources (none in BtS and, so far, also none in AdvCiv) are placed in the "Buildings" box instead of "Allows".

Tbd.: Should perhaps guarantee the extra move until the end of a turn if the resource was available at the start of that turn.

Use the ability for Industrial-era ships: allow them to be trained with Coal, but make them faster with Oil. Uranium should then be removed as an alternative requirement for Destroyer.

Rationale: Until AdvCiv 0.97, I had used this ability to represent auxiliary steam engines that were

installed on ships of the line in the early 19th century ([Wikipedia](#)), but, when I realized that Frigate is too slow in BtS, this complication became unnecessary because, with Frigate at speed 5, Ship of the Line can have speed 4 without any special justification.

The details of what it means that a unit has access to a resource are a little messy. The trade network doesn't extend onto hostile tiles, but I don't want units to lose speed in hostile territory, especially not immediately upon entering. Relying entirely on the availability in the capital could lead to situations where a unit sits right on a resource tile in a friendly Fort or city, and still doesn't get the movement bonus when the tile isn't connected to the capital. With the current solution, a unit can, in theory, still lose its extra moves upon entering hostile territory, but this should be rare.

No strategic AI evaluation: I want to use this ability only for giving naval units (which are already fast) one extra move; i.e. the significance is going to be minor.

906	AdvCiv	K-Mod	BtS

Submarine	24 strength
	6 first strikes
	7 moves
	cost 210
	req. Rocketry, Radio,
	  Uranium
	renamed to
	  "Nuclear Submarine"
		24 strength
		+25% attack
		6 moves
		cost 150
		req. Radio,
		  Oil or Uranium
			24 strength
			
			6 moves
			cost 150
			req. Radio,
			  Oil or Uranium

Attack Submarine	24 strength 6 first strikes 6 moves cost 160 req. Radio	30 strength +25% attack 7 moves cost 200 req. Rocketry, Radio +50% vs. Submarine	30 strength req. Rocketry, Radio 7 moves cost 200 (180 in v3.13) +50% vs. Submarine
Switched graphics of (Nuclear) Submarine and Attack Submarine			
Battleship	250 cost	225 cost	225 cost
Destroyer	200 cost ignores first strikes	200 cost	200 cost

	AdvCiv	K-Mod	BtS 3.17-3.19	BtS 3.01-3.13

Stealth Destroyer	270 cost
	30 strength
	1 first strike
	sees stealth ships
	  and submarines
	ignores first strikes
	can load 1 Missiles
	30% chance to
	  intercept aircraft
	requires Stealth
		220 cost
		36 strength
		2 first strikes
		sees stealth ships
		
		requires Stealth
		  and Robotics
			220 cost
			30 strength
			2 first strikes
			sees stealth ships
			
			requires Stealth
			  and Robotics
				250 cost
				30 strength
				16% bombard rate
				sees stealth ships
				
				50% chance to
				  intercept aircraft
				requires Stealth
				  and Robotics

Rationale: The BtS submarines were too weak against Battleship and the role of Attack Submarine (hunter-killer) was too narrow. Now Attack Submarine is the ordinary submarine unit for the World Wars, and Missile Submarine a more expensive unit representing the cruise missile and ballistic missile subs of the Cold War. First strikes match the flavor of torpedoes and invisibility well; consistent with first strike on Stealth Destroyer. Strength 25 and 5 first strikes would have a nicer

symmetry with the 50% withdrawal ability, but that would make Submarines a bit too effective against Destroyers, which, as anti-submarine units, get to ignore first strikes.

Increased the cost of Battleship in order to further weaken that unit against submarines. The BtS cost is also too close to that of Destroyer.

Stealth Destroyer isn't needed as an efficient combat unit, and this would also be unrealistic. The few stealth destroyers that have been manufactured are classified as multi-role guided missile destroyers, and have strong anti-air and anti-submarine weapons. These additional abilities make the Stealth Destroyer an expensive Swiss Army knife. Removed one first strike to make the multiple first strikes of the Submarines more special. The Robotics requirement seems nonsensical and makes the unit very difficult to access.

Credit: Dawn of Civilization lists Stealth Destroyer's submarine detection as a bugfix in its list of features: "*Stealth Destroyers detect Submarines as (presumably) intended*"

See also: 028 allows Submarines and Stealth Destroyers to defend weaker visible units.

I've only skimmed through this thread (title: "Submarine Confusion"), but I think the participants mostly agree that BtS has its two submarines confused.

164 makes the Blitz promotion easier to access (also for post-Renaissance ships).

Tbd.:
Destroyer is still a bit weak against Submarines. May have to give it an explicit combat bonus against Submarines (instead of relying entirely on first strikes vs. first-strike immunity), or give Submarines an Ocean attack bonus.

Would be nice to show the abilities shared by the two submarines more compactly in help text and Civilopedia:
"Submarine (invisible, reveals Submarines, can dive under Ice, can explore rival territory)"

Lower the strength of all post-Combustion ships a bit; they don't need to be *that* much more powerful than Ironclad.

Air recon shouldn't reveal Submarines so easily.

Would like to give Stealth Destroyer two missile slots, but so long as two Tactical Nukes can eliminate any stack, this seems a bit too dangerous.

Missile Cruiser needs work. Want to move it to Rocketry, but will first have to make room by moving the spaceflight stuff to Satellites.

907: Uniques	*AdvCiv*	*BtS*	*before patch 1.52*
907a Praetorian	strength 7, cost 40, starts with March named "Legionary"	strength 8, cost 45, named "Praetorian"	strength 8, cost 40 named "Praetorian"

	Rationale	Clearly intended to be easy to recognize as powerful even for new players, and it worked, but this mod isn't for new players, and the unit is toxic for game balance. Supplants all pre-Renaissance units plus Musketman. Because it's so obvious, I'm nerfing this unit before all others.

<table>
<tr><td>Rationale</td><td>Clearly intended to be easy to recognize as powerful even for new players, and it worked, but this mod isn't for new players, and the unit is toxic for game balance. Supplants all pre-Renaissance units plus Musketman. Because it's so obvious, I'm nerfing this unit before all others.

7 strength for 40 would be fine, I think, but bland. March is a promotion that doesn't give the unit more raw power, but is still useful, and a good fit flavor-wise as the Roman army was known for its forced marches. (Trivia: The BtS manual incorrectly claims that Phalanx starts with March.)

As for the name change (only in the English game text), Soren Johnson mentioned in a Twitch video that "Praetorian" was chosen over Legion(ary) to avoid repeating too many names from Civ3, and seemed regretful about this. Legionary is obviously a better fit for the BtS unit, and also fits well with my changes.

Looks like karadoc had also been considering a name change; there was a text key commented out (now deleted) in the K-Mod game text file that would name the unit "Legion".</td></tr>
<tr><td>See also</td><td>131 gives Legionary Attack City as its default AI type.

CFC thread discussing overpowered unique units.</td></tr>
</table>

	AdvCiv	K-Mod	BtS
907b Quechua	cost 15, starts with Combat I, named "Quechua Warrior"	cost 20, starts with Combat I, +100% vs. Archer, named "Quechua"	cost 15, starts with Combat I, +100% vs. Archer, named "Quechua"
Rationale	The BtS unit is far too powerful, and I don't think the K-Mod nerf really changes that. And players who aren't willing to take advantage of the anti-Archer bonus end up with a unit that is worse than a regular Warrior. "Quechua Warrior" because Quechua is just an ethnic group (like Oromo). Trivia: Free Combat I was added by the Warlords expansion.		
Tbd.	Want to turn Holkan into the unique Warrior because such an early unique unit makes more sense for an early civilization like the Maya than for the Inca. Free Combat1 could still encourage rush strategies, so I'd rather give the unit an ability that lets it gain XP faster. The Quechua could then become a unique Maceman with lower combat strength and easier tech requirements than a regular Maceman.		

	AdvCiv	BtS
907c Fast Worker	2 moves, ignores terrain movement costs	3 moves
See also	Version 3.0.0.5 of the "Rebalance the Realms" (RtR) multiplayer mod gives Fast Worker the Mobility promotion. RealmsBeyond.net (under "Civs"). The current version of the mod (5.1.1.0) instead replaces Fast Worker with a "Urukku Pikeman" (urukku presumably refers to wootz steel). The Close to Home fork of RtR also uses the Mobility promotion. The Tides of War multiplayer mod gives Fast Worker a higher production cost.	

Rationale	Widely regarded as one of the best unique units. This is a bigger nerf than I would like, but I really think that no pre-modern land unit should have more than 2 moves, and it seems that the most skilled players appreciate Fast Worker most for being able to move and chop in a single turn – an ability that my change preserves. Ignoring movement costs vs. Mobility makes a difference only on Forest/Jungle Hills. Most players probably aren't aware that the movement penalties stack in those cases, and I don't think they should be aware. Also, free Mobility would be an even weaker ability than the one I use.	
Tbd.	The name "Fast Worker" doesn't fit well for the new ability, and it's a lame name in any case. It's also not clear why India should have any such unit at all; the reasons given in the Civilopedia are at best contrived and at worst an offensive stereotype.	

	AdvCiv	*BtS*
907d Skirmisher	strength 4, 0-1 first strikes	strength 4, 1-2 first strikes
See also	Recent CFC thread with some opinions on Skirmisher: link Multiplayer balance mods: The minimalist Tides of War multiplayer mod sets it to 0 first strikes. Rebalance the Realms adds a 40% attack penalty against Archers. Close to Home leaves the unit as it is in BtS.	
Rationale	One extra strength on an early unique unit is generally too much without any drawback. Since Archers aren't good at attacking cities, it might've been enough to remove only the first strike chance, especially considering that Mali already suffers from a nerf to the Financial trait (see change 908a below); hard to say. Mali still does well in all-AI games.	

907e (added in v1.0)	*AdvCiv*	*BtS*
War Chariot	strength 5	strength 5, immune to first strikes
Immortal	can receive defensive bonuses, +25% combat vs. Archery units, immune to first strikes	can receive defensive bonuses, +50% vs. Archery units

See also	Multiplayer balance mods: [Tides of War](#) gives War Chariot only 4 strength and 2 first strikes (and no immunity); no change to Immortal. [Rebalance the Realms](#) gives War Chariot a 25% penalty vs. Archer(y?) and new abilities altogether to Immortal. Close to Home leaves both as in BtS.
	[CFC report](#) about an AdvCiv 0.97 game on Emperor difficulty where the player immediately conquers two neighbors with War Chariots, then agrees with me that War Chariot and Immortal are arguably too powerful.
	[CFC thread](#) about trying to beat AdvCiv 0.98 on Deity through War Chariots. Doesn't really work, but still suggests that War Chariot is the most effective early rusher with Quechua already nerfed at that point. [Related post](#) where user drewisfat indirectly refers to War Chariot as "super OP."
	[CFC post](#) (end of the post) arguing against AdvCiv nerfing Immortal, seeing that Persia's Financial trait was already nerfed.
	2010 [CFC thread](#) comparing War Chariot and Immortal
Rationale	About War Chariot, there seems to be pretty broad agreement that it's among the most powerful units, if not in multiplayer, then at least against the AI. It's not so clear how much War Chariot should be nerfed and whether Immortal is just as problematic.
	Focusing on both units' potential as early attackers against lightly defended AI cities, dcor's combat odds [calculator](#) (download no longer available) tells me that a regular Chariot has 8.3% victory odds (i.e. 18.3% survival) against a fully fortified Archer in a city with 20% culture defense. In BtS, both War Chariot and Immortal do about 20 points better: War Chariot has 28.3% victory odds, Immortal 27%. With the changes that I've made, the victory odds become 22.8% for War Chariot and 24.1% for Immortal. With a Combat I promotion, both units land again around 27-28%. Do I want to nerf War Chariot further by slapping +5 on the production cost? Can't decide, so I'm going to leave it as in BtS.
Tbd.	The Immortal should be turned into a completely different unit, namely a Spearman (graphics from e.g. Realism Invictus) with a national unit limit. The current stats are complicated, weird and difficult to connect with Persian history.

	AdvCiv	K-Mod	BtS
907f Panzer	+50% vs. Armored 2 first strikes	+50% vs. Armored 1 first strike 1 first strike chance starts with Flanking	+50% vs. Armored
Rationale	To streamline the K-Mod change. Normally, I would've reverted the K-Mod change entirely because I don't think buffing unexciting uniques is worth the added complexity (and buffing those that only appear in the late game is a bit of a lost cause), but the K-Mod change has some historical validity too I think: The Panzer V (arguably) depicted by the 3D model had a particularly high range.		

	AdvCiv	BtS	Vanilla prior to v1.61
908: Traits			

908a Financial	+1 commerce in tiles with a natural yield* of at least 2 commerce or a total yield of at least 3 commerce.	+1 commerce in tiles with at least 2 commerce.	+1 commerce in tiles with at least 2 commerce. Faster production of Bank

* By "natural yield" I mean the yield from terrain, feature, river and hill. Yield from improvement, building, trait or civic is not part of the natural yield.			
See also	031 and 053 reduce AI found values counted for rivers (with the Financial trait change in mind). Several mods, e.g. Rebalance the Realms (RtR), raise the extra yield threshold from 2 to 3, which is easy to do through XML. The RtR fork Close to Home preserves the threshold of 2 on water tiles. CFC post lamenting how much the Financial trait is nerfed by AdvCiv. (As the next post points out, Organized trait being superior to Financial would be due to the map size and rapid military expansion; so be it.) Reddit thread discussing the change (and arguing for a Bank discount from Financial).		
Rationale	The Financial trait is easily the most potent trait in the hands of the AI, to the point that games with several Financial leaders get far ahead of the historical time line, and, unlike most other top-tier traits, players can't help but exploit it. Plus, the K-Mod change to Serfdom makes Financial even more powerful.		
Tbd.	The nerf perhaps goes a little bit too far. Restore the pre-1.61 building discount? I think I'd like a discount for Market better because the Bank discount coincides with the English unique building (Stock Exchange) and two of the English leaders have the Financial trait. The Bank discount also seems more powerful. On the other hand, change 911b already reduces the base cost of Market, Bank is more flavorful given the name "Financial", and the Bank discount had been the designers' original intent. The patch may have been intended both as a nerf of the Financial trait and of the English civ – Redcoat was still a very powerful unit at that time (but no longer is since the Warlords expansion).		
Config	Through a new XML tag `ExtraYieldNaturalThreshold`. Not very flexible, but at least this allows mod-modders to restore the original trait effect (by using the original `ExtraYieldThreshold` tag instead the new tag). (Before v1.0, I had simply given the old tag the new semantics.)		
	AdvCiv		*BtS*
908b Creative	• Cities start at Fledgling culture level (including conquered cities once occupation ends). • +1 culture rate in all cities. • Faster production of Library and Theater.		• +2 culture rate in all cities. • Faster production of Library, Theater and Colosseum.
See also	Chapter about changes to the culture system		

Rationale	Creative is already a powerful trait in BtS and easy to use; the changes to the culture system make it too powerful, considering that other powerful traits get nerfed. It's also frustrating to have a Creative leader as a neighbor; usually not economical to compete with them for border tiles. Just halving the culture rate bonus goes too far, so the (one-time) free culture is supposed to compensate to an extent. Also removes the rather pointless 5-turn waiting period until the borders of a new city expand, thus also communicating more clearly what the Creative trait is useful for (expanding borders). One might argue that the 5-turn delay is an important check on the early-game access to strategic resources – but one can always get that by settling adjacent to the resource (or even on top of it). Not sure if the ability should really apply to conquered cities. Seems more consistent with the old ability this way (the +2 culture per turn applies to all cities – so long as they're not under occupation).
Tbd.	Getting access to the outer ring 5 turns earlier in new and conquered cities might be more powerful than I realize. One could exclude conquered cities if the current ability is too powerful. If it's not powerful enough, one could restore the discount for Colosseum. I had removed that in AdvCiv 0.99 as a minor balance change that, in my view, turned out to be insufficient. The flavor doesn't really fit, but at least it would give players a reason to construct Colosseums sometimes.
Config	Can restore the BtS abilities through `Civ4TraitInfos.xml` and (Colosseum discount) `Civ4BuildingInfos.xml`. In `Civ4TraitInfos.xml`, the new tag `iFreeCityCulture` gets displayed as "cities start at such-and-such culture level" if the free culture is exactly equal to the threshold of a culture level; however, it's also possible to set any amount of free city culture.

Also reduced the culture rate of Terrace from 2 to 1.

Rationale	Terrace is regarded as a light version of the Creative trait; don't want it to become actually more powerful than the Creative trait. Also, it's arguably the most powerful unique building in any case and the culture changes in AdvCiv are making it even better.	
See also	[201](#) deals with other changes to building culture rates.	
	Hippodrome grants 1 happiness per 10% culture; Theater and its other unique replacements grant 1 happiness per 20% culture.	Hippodrome: 1 happiness per 5% culture; other Theater-class buildings: 1 happiness per 10% culture.
	Colosseum and its unique replacements grant 1 happiness per 10% culture.	1 happiness per 20% culture.
Rationale	To make up for the lost production discount from the Creative trait. Colosseum is already a pretty weak building in BtS, don't want to make it even less appealing overall. (That said, a substantial buff would make Temple too unappealing.) Between Colosseum and Theater, it's more intuitive to give Colosseum the stronger happiness ability because Colosseum is all about happiness and Theater primarily about culture. And using the culture slider against (temporary) problems with unhappiness feels like a pretty faithful representation of Roman ludi. Weakening Theater a little bit doesn't hurt.	

See also	200 reverts the K-Mod changes to Colosseum (25% culture, +20 production cost); 251 lowers the Legendary culture threshold instead.

908c Philosophical	+80% GP birth rate, faster production of University	+100% GP birth rate, faster production of University
Rationale	Widely acknowledged as a top-tier trait, and I would say the one most commonly identified as *the* most powerful trait by Deity players (but this would be a close call). Nerfing Financial was in my estimation more important for non-Deity players (Deity players don't really play this mod anyway) and the AI, but I don't think I can leave Philosophical alone after nerfing Financial. Not sure how much to reduce the rate. Anything less than 100% is already a substantial nerf because rounding errors come into play. 67% would make very clear that a single specialist will generate 2 extra GP points. However, this is also clear enough with 80%: 2.4 isn't going to get rounded up, i.e. even if a player isn't aware that rounding down is always the rule, they can't go wrong. For two specialists, there is still no difference between 67% and 80% in terms of power level (both round down to +4), but, above that, 80% is more powerful and some base birth rates below 6 also round more favorably with an 80% modifier. Steps of 25% is what the game normally uses, but I'm not aware of 75% being used for anything except SDI, and going below 80% would really seem like too big a nerf.	

909a: Late game units	*AdvCiv*	*K-Mod*	*BtS*
Machine Gun	+25% vs. Mounted +50% vs. Gunpowder	+10% vs. Mounted +50% vs. Gunpowder	+50% vs. Gunpowder
Rationale	At 10%, Cavalry still was too effective against Machine Gun. Not a gameplay problem, but pretty unrealistic. It might be that charging machine guns on horseback is a better idea than doing so on foot, but it's still a bad idea. Also, 10% is an unusual modifier.		
See also	CFC discussion Post by karadoc contemplating a 25% bonus vs. Mounted.		
Tbd.	Cavalry is also too effective against Infantry. Machine Gun shouldn't be immune to collateral damage (none of the Siege units should be, but especially not Machine Gun).		

910	Era tech cost modifiers adjusted (increased overall). These modifiers were added by BBAI. There's one per era, and it changes the costs of all techs in that era. K-Mod has tweaked them, and eventually (version 1.45) tied them to the inflation rate, a change not adopted by AdvCiv.
Rationale	To match the historical dates of discovery better. E.g. Gutenberg's printing press was invented around 1450, so the Printing Press tech shouldn't be discovered much earlier in a typical game of Civ.
Config	Set in `Civ4EraInfos.xml`

See also	<u>251</u> adjusts the AI research speed based on difficulty. <u>140</u> adjusts tech costs based on map size. <u>308</u> lowers tech costs when Raging Barbarians is enabled. <u>550d</u> lowers tech costs when playing without tech trading.
Tbd.	Perhaps reduce the travel duration of the spaceship. 10 turns is a long time that late in the game and it's not like the spaceship launch comes as a surprise.

AdvCiv	BBAI
The era tech cost modifier is applied to the tech cost.	Applied to the every civ's research rate.

Rationale	More transparent this way, as (modified) tech costs are prominently displayed.

AdvCiv	BtS
No research speed bonus for possessing a single prerequisite tech. Reduced era tech cost modifiers a bit, but it seems that various AI changes have accelerated the AI tech pace enough to almost make up for the -20% research, so the AdvCiv cost modifiers aren't that different from those in BBAI and K-Mod.	BtS grants a 20% research speed bonus for every known prerequisite tech, even for the first prerequisite. (Only prerequisites represented by an arrow count in this context; called "or prereqs" in the code.) As a BBAI comment observes, "this [effectively reduces] the cost of most techs on the tech tree [by 20%]." Not all techs because e.g. Astronomy has no incoming arrow, and the six techs in the very first column have no prerequisites either. BBAI has added an XML setting for disabling the speed bonus for the first known prerequisite.

Rationale	I want to leave the research times (turns to research) mostly unchanged because the original developers and modders have balanced the game around those research times. Whether a tech has an incoming arrow or dangles in the air is a technicality that should affect neither research speed nor cost.
	Again, a tech cost adjustment is more transparent than a research speed adjustment.
	Note that research generated through the discover ("bulb") ability of a GP is unaffected by research rate modifiers, so turning a research rate bonus into a tech cost discount makes bulbs more powerful. However, this is pretty much evened out by the change to the era-based modifiers above.

Config	The BBAI setting is `TECH_COST_FIRST_KNOWN_PREREQ_MODIFIER` in `TechDiffusion_GlobalDefines.xml`.
Tbd.	Can't continue increasing tech costs; will eventually have to decrease the AI bonuses from difficulty level instead.

AdvCiv	BtS

Help text for technologies that the active player can research shows (also on the research bar) the research modifier for the given tech unless it's 0, and a breakdown of the modifier. Since all the static factors are now applied to the tech cost modifier instead of the research modifier, the only components in the breakdown are "from diffusion" (BBAI diffusion if enabled, otherwise the BtS research bonus from teams that already know the tech), "from tech prereqs" (more than one OR prereq) and "from teammate with same research" (see 156).	The modifier comes only from OR prereqs (see above) and diffusion. It's not shown anywhere.
Rationale	Anything related to research speed is an important statistic. And need to make players aware of change 156 (in team games).
On the research bar and the choose-research popup, help text shows when one researchable tech is going to speed up another. E.g. "Priesthood – Speeds up Writing by 20%" when Animal Husbandry already makes Writing researchable. Or "Code of Laws – Can be sped up through Priesthood by 20%" when Priesthood not yet discovered.	Doesn't mention the link to Writing at all when Writing is already enabled by Animal Husbandry. No reminder about possible speed-ups.
Rationale	I had at first thought that the "can be sped up" text wouldn't be needed, but I've come to realize that it's an important reminder.
See also	CFC post requesting the "can be sped up" text. The BULL option "Speeds Up info" shows essentially the same information, but I've written my own code because the text has to match the change in the blue box above and the one below.
When hovering over a tech (as above), the techs enabled by that tech are listed in the following order: 1. Techs that will be immediately enabled before techs that have another missing requirement. 2. Within these categories, techs are ordered by cost (ascending). Example: "Mathematics – Leads to Construction, Currency and later Calendar, Music, Civil Service"	Ordered by tech id, which is determined by the order in `Civ4TechInfos.xml`, i.e. arbitrary. "Mathematics – Leads to Music, Civil Service, Calendar, Currency, Construction"
Tbd.	Would be nice to do something similar for enabled units (also taking into account resources?) and buildings.
Changed the game year computation on Normal speed: After AD 1400, the game continues in steps of 10 years for 5 turns longer than in BtS. This is compensated by shorter steps later in the game.	
Rationale	The (AI) civs kept getting ahead of the historical time line in Renaissance, even after I had increased Renaissance tech costs by 25%. Now the game year counts a bit faster during Renaissance.
Config	`Civ4GameSpeedInfos.xml`

Tbd.	Not sure if the other game speed settings need a similar adjustment. In some quick tests, the tech progress looked OK.
	Reduce the total number of game turns to something like 480? The turn limit for Time victory comes so late currently that it's practically irrelevant.
The tech cost modifier based on map size increases in small increments (5 percentage points) for the small map sizes, makes a jump from Standard (115%) to Large (137%) and a smaller jump from Large to Huge (150%).	Duel size has 100% tech cost, from there it's +10% for each size level.
Rationale	Trial and error. I don't understand why Large maps lead to much higher research rates than Standard size.
See also	[140](#) deals with other map-size adjustments.
The map size modifier is ignored for Ancient-era techs.	Applies to all techs equally.
Rationale	In the Ancient era, the map size doesn't normally affect the pace of expansion, and the number of civs met doesn't matter yet because tech trading doesn't become available until the Classical era.
Credits	Got the idea from the BASE mod (v6.3 [changelog](#); in German). It sounds like BASE might also exempt Classical tech.
Tbd.	Perhaps apply the modifier half (or times 75%) for Classical tech.
If tribal villages are disabled, the cost of Ancient techs is reduced by 5%.	
In Always-Peace games, tech costs are increased by 5% after the Ancient era.	
Tech costs are adjusted to the sea level setting (if the map has one): +20% with Low sea level, -15% with High sea level.	The sea level affects the space available for cities.
See also	[137](#): impact of sea level on the number of supported players.
Config	Added a tag to `CIV4SeaLevelInfo.xml` for the sea level modifier. The tag is called `iResearch` because that's how BtS usually names tech cost modifiers.

911	Misc. building changes	
911a	*AdvCiv*	*BtS*
Spy specialist slots	2 at Courthouse, 1 at Jail.	2 at Jail, 1 at Courthouse.
See also	[310](#) lets the Great Wall generate Great Merchant points instead of Great Spy points.	
Credits	Proposed and advocated by crullerdonut (CFC [post](#); after the 3rd quote box)	

Rationale	So that players who want to conduct active espionage missions in the midgame don't have to wait until Constitution. I think it's fair enough if espionage-focused strategies only become viable toward the Industrial era, but it doesn't hurt to enable them earlier – so long as this doesn't mean that the espionage systems becomes more difficult to avoid for players (such as myself) who dislike it. A concern that Courthouse may become the go-to building for early Great Person "farms" is hopefully addressed by the changes to Market and Grocer below.	
	An alternative consideration was to replace the +25% espionage at Castle with the Spy specialist slot from Jail. Considering how late Castle becomes available and that it's a very situational building, this would've been quite an extra hoop to jump through.	
	In BtS, it was possible to get an espionage-based economy started through an early Great Wall. This is no longer possible in AdvCiv, so the extra Spy slot at a non-wonder building is supposed to compensate for that.	
See also	CFC post by drewisfat. The last paragraph (written prior to the Courthouse change and without having played AdvCiv) argues that adopting an espionage economy is (essentially) only ever worthwhile in the early game due to AI spam of espionage buildings and spies in the late game. I don't think the AdvCiv AI behaves that way. K-Mod has revised much of the espionage AI code and AdvCiv has fixed a K-Mod bug that had caused the AI to run the Big Espionage strategy far too often, and has made further tweaks that discourage AI investments into espionage; cf. 120.	
	These changes in mind, I hope that the Spy slot at Courthouse comes early enough to make an espionage economy a consideration sometimes.	
911b - Market, Grocer	130 production cost	150
Rationale	They're overcosted in BtS, but so are many buildings, in particular other happiness buildings. I don't want to adjust too many building costs (avoid overwhelming players with the volume of rule changes), so I'm making only a moderate adjustment that doesn't make Temple and Colosseum look too much worse.	
	The impetus for changing Market and Grocer at all is that I want to avoid making Courthouse a superior choice for Great Person "farms." Don't want players to feel that they have to engage with the espionage system beyond passive missions.	
911c - Aqueduct, Baray	90 production cost	100
Rationale	It's pretty clearly overcosted in BtS – worth +2 food at best – and for an early building that is almost iconic for the Civ series, that's more of a shame than e.g. Customs House being (even more) overcosted. Also, this minor, self-explanatory cost change should demand very little attention from players who are new to AdvCiv.	
	A cost of 80 could also be justified (that what the Close to Home multiplayer mod does for example), but I think 90 should be enough to make Aqueduct worthwhile when the extra health is really needed, and I don't want it to feel too cheap considering that aqueducts were pretty big construction efforts in reality. (In that respect, Colosseum also feels too cheap at 80 production.)	
	I don't want to buff Hammam, so I'm leaving that unique building at its BtS cost of 100.	

912	Civics		
Tbd.	Lots of problematic civics … Some projections in hover text would be helpful. E.g. number of worked and owned Farms, Plantations and Towns for Serfdom. Related CFC post (near the middle)		
	AdvCiv	*K-Mod*	*BtS*
912a Serfdom	+1 commerce from Farm, Plantation -1 commerce from Town +50% Worker speed Medium upkeep	+1 commerce from Farm, Plantation -1 commerce from Town +50% Worker speed Low upkeep	+50% Worker speed Low upkeep
Rationale	The switch to Emancipation is a bit too painful in the K-Mod version. At Low upkeep, Serfdom is usually still less powerful than Slavery, but that's not a good baseline, and, for the AI, I'd say Serfdom is more useful than Slavery even with Medium upkeep.		
See also	131b assigns a negative AI weight to Feudalism because the tech is too popular now.		
912b Pacifism	military cost 1 gold; only 0.5 gold for the AI	military cost 0.5 gold	military cost 0.2 to 1 gold depending on difficulty (e.g. 0.2 on Settler, 0.5 on Noble, 1 on Deity)
Rationale	It's good that K-Mod decoupled the cost from the difficulty setting because the difficulty-based modifier was undocumented and needlessly complicated. However, the K-Mod change made Pacifism cheaper on all difficulty settings above Noble, and Pacifism had already been one of the more powerful civics in BtS, even on Deity. The AI used Pacifism rarely in K-Mod, and the AdvCiv change might've made the civic unusable for the AI. Therefore, the AI supply cost modifier now applies to the Pacifism cost. This should be not nearly as confusing as the BtS approach because the human cost is always the same, and the AI cost is simply 50% of that. Since the AI needs far more units than the human players to guard its cities, I think it's fair to place a 50% discount on anything that punishes a high unit count.		
See also	250d makes the AI supply cost modifier truly independent of difficulty. Git commit message about the K-Mod change to Pacifism		
Civics screen shows the cost per unit without inflation (i.e. shows just 1 gold) and shows the current total cost (hypothetical if not currently running Pacifism) including inflation.		Cost per unit shown including inflation. No total shown.	Cost per unit without inflation, no total.
912c	*AdvCiv*	*BtS/K-Mod*	
Hereditary Rule	1 happiness per 2 military units +25% happiness from resources Low upkeep	1 happiness per military unit Low upkeep	
Config	Can restore the BtS ability in `Civ4CivicInfos.xml`. If `iHappyPerMilitary Unit` is set to 2, the game treats it as 1 happiness per (one) military unit.		

Rationale	The BtS ability is fiddly, encourages players to train lots of Warriors and never upgrade them (or generally to train more units than needed), makes it too easy to grow cities without buildings and is a poor fit flavor-wise (would fit for a military dictatorship). This ability should arguably be removed from the game entirely, but the replacement that I have in mind would require some other balance changes that I don't want to make at this point.
	I had to come up with a temporary solution after change 036 (AI trades away non-surplus resources if it doesn't need them) because there were too many civs offering happiness resources during the second third of the game and too few civs willing to import them.
	The bonus to resource happiness should make Hereditary Rule better than Representation at least in some (rare) circumstances, namely when a civ has a lot of luxury resources (and the matching buildings) and needs even more happiness (due to war weariness I guess). The bonus also means that civs can normally get 2 extra happiness out of Hereditary Rule without making a particular effort, namely by having at least 4 luxuries and station-ing a second unit.
912d Slavery	New game option "No Slavery" prevents human civs from adopting the Slavery civic but allows cities with a Sacrificial Altar to sacrifice population regardless of civics. The option also causes human Granaries (and Terraces) to store 1.2 times as much food.
See also	160 lets Granary store only 40% food; i.e. the "No Slavery" option negates that balance change.
Tbd.	A replacement civic; see future gameplay changes, specifically this file. The Kremlin will also need a new ability eventually.
Rationale	The balance issues with the hurry ability could probably be solved by allowing only 1 citizen to be sacrificed at a time and by basing the production yield on the city size, but the ability will still be complicated, tedious to use (practically requiring the noisy can-hurry alert to be enabled) and highly unrealistic (less so with Sacrificial Altar).
	So long as I haven't implemented a civic to replace Slavery with, disabling Slavery needs to be optional. Treat the "No Slavery" option as an extra challenge for the human player for now.
	Of course one could simply not use Slavery, without the need for a game option, or use it only to train defenders in the event of a surprise attack; the game option is mostly intended to raise awareness.
Config	The Granary adjustment is hardcoded in the DLL (and affects any building that causes food to be kept after growth). It gets applied to the human proxy civ during AI Auto Play (in order to avoid problems with inconsistent city data), so, ideally, "No Slavery" should not be enabled for all-AI games.
	Can hide the "No Slavery" option in `Civ4GameOptionInfos.xml`. For a simple balance change to Slavery, one could lower `iProductionPerPopulation` from 30 to 20 in `XML\GameInfo\ CIV4HurryInfo.xml` (Vanilla Civ 4 assets) as recommended here by Fippy. The AI should be able to handle it.
See also	121 about how the AI uses Slavery.
912e	*AdvCiv* *BtS*

On Quick speed, up to three civics can be changed in one turn of anarchy.		Up to four.
Credits	From Civ 4 Reimagined ([Git commit](#))	
Rationale	While a turn of anarchy is brutal on Quick speed, allowing nearly all civics to be changed still seems strange.	
912f	*AdvCiv*	*BtS*
Can set cottage growth modifiers that are not a multiple of 100. Negative growth still isn't possible; no growth (-100% modifier) is possible.		The Emancipation ability only works for positive multiples of 100 because the game counts the time that an improvement has been worked in whole turns. Letting a civic disable Cottage growth causes a crash in K-Mod AI code. (And any negative modifiers get rounded down to 0 growth.)
Rationale	Doesn't make any difference in AdvCiv, but it's an unexpected problem for mod-mods. A downside is that this change increases the memory footprint of a CvPlot object by 2 byte (to be able to count improvement turns at a higher precision).	

advc.mxc	Mixed continents map script
See also	chapter [Mixed Continents](#) [165](#) decreases the grid dimensions of maps with long coastlines a bit.
Rationale	I've been looking for a map where water matters and that still (unlike Archipelago) allows most players to place some 5 core cities on their home continent and to wage some land wars, and that isn't as predictable as Inland Sea or the Mediterranean setting of Tectonics (or Eyeball Planet or Caldera). Scripts that place the sea in the middle also give everyone at most two neighbors. Mixing together small and medium-sized continents (some of them merging into bigger continents) seems to work pretty well; "Big and Small", "Medium and Small" and "Not Too Big or Small" don't allow that, but "Continents and Islands" does.
Tbd.	Testing – I've looked at maps generated for various settings and their statistics, but haven't played even one full game on the map. Translations for some of the options (should do at least German for completeness' sake). To keep the script self-contained (portable), the hardcoded English texts should be used as a fallback.

Mixed Continents	*Continents and Islands*
Renamed some options to reflect that this isn't necessarily a map with continents *and* islands.	Options named "continent size", "island size", "islands mixed in".
Allow all landmass sizes for both regions.	"Big and Small" allows only small ("snaky"), normal and massive continents for the first region and tiny and normal islands for the second. "Continents and Islands" allows all sizes for the second region, but no islands for the first.
Rationale	I don't think a "Small & Small" map (without any continents) is going to play much better than Archipelago, but there's no harm in allowing it.

The maximal number of tiny islands is based on the map size.	Regardless of the landmass settings, some tiny islands are placed on the map. The number is chosen at random between a minimum of 1 and a maximum of 4.
Base the hill grain value of each region on that region's landmass size: spread the hills out a bit more on tiny islands, normal islands and snaky continents.	Hills are spread out a bit more in region 2 than in region 1.
Rationale	In reality, the size of a landmass doesn't determine the ruggedness of its relief; however, for gameplay, it's important that it's not too difficult to give cities access to a couple of hill tiles. On small landmasses, a higher portion of the tiles in city radii are water tiles, which can't be hilly. So it's sensible to adjust the hill grain to the landmass grain value; the problem with the "Continents and Islands" code was only that region 2 doesn't necessarily contain small landmasses.
Use the default latitude thresholds for snow and tundra.	Custom thresholds that are higher than in BtS, i.e. less Tundra and Snow.
Rationale	The BtS thresholds had been too low – or at least unrealistic –, but I've already addressed that through `CvMapGeneratorUtil.py` as part of tsl.
Tbd.	The K-Mod changelog says about "Not Too Big or Small" that "*[z]ones are now more likely to have a buffer at the north and south poles.*" This buffer should perhaps be shrunk to match the AdvCiv tundra and snow latitude thresholds.
"Add Plains" and "Reduce Desert" options removed.	Options "Terrain Clumping", "Add Plains", "Reduce Desert" added.
Rationale	The plains and desert options aren't useless but pretty random (Why can't Desert be *added*? What about all the other terrain types?) and thus confusing. Better to adjust terrain frequencies through the Climate option or adding a new climate type. (In fairness, the plains frequency can't be adjusted through climate.)
	"Terrain Clumping" seems a bit less arbitrary, so I'm -tentatively- keeping that one.
Config	Shouldn't be too difficult to re-enable the removed options in `Mixed_Continents.py`; just commented out.
World wrap option added.	
Rationale	No reason why this script shouldn't have that commonplace option.
See also	kekm.32 adds a world wrap option to "Not Too Big or Small".
Mixed Continents	*Continents and Islands*
The sea level setting is used and has approximately the same effect on the sea/ land ratio as in most other map scripts (regardless of the region-overlap setting).	"Big and Small", "Medium and Small" and "Continents and Islands" all ignore the sea level setting.
"Add Water" option removed.	"Continents and Islands" has an "Add Water" option instead.
Rationale	Ignoring the sea level without so informing the player is terrible. May have been an oversight, or Sirian thought that the sea level should be controlled through keeping the two regions separate or mixing them together (the latter results in more land). Of course it's better to have a separate control.

Config	The "Add Water" option is commented out in `Mixed_Continents.py`. If the sea levels aren't fine-grained enough, more elements can be added to `Civ4SeaLevelInfo.xml`. I haven't tested it, but I think the mod should be able to handle additional sea levels. Replays will no longer be BtS compatible though (see [106i](#)).	
AdvCiv		*K-Mod*
"Not Too Big or Small" takes the sea level into account.		Ignores sea level. No way to adjust the land/ sea ratio.
Rationale	Since "Not Too Big or Small" doesn't allow regions to overlap, this change was easy to implement; strange that karadoc hadn't done this already.	
See also	Kek-Mod has added an "Adjust Water Percent" option to "Not Too Big or Small". Git [commit](#)	

advc.gfd	GameFont display
Credits	By Nightinggale, adopted without any noteworthy changes. Original mod component: [CFC link](#)
Config	Shift+Ctrl+F1 opens the GameFont display window
Rationale	I use the `GameFont.tga` from BULL and have no plans of extending it, so I probably won't need this, but it could be handy for mod-mods and was very easy to integrate.

test	Temporary changes for debugging and testing are marked with `advc.test.` or `advc.tmp.`
Tbd.	Most of the test code should be removed eventually. Currently, there are quite a few branches for recovering from invalid call arguments that don't ever seem to occur. Those should be turned into assertions. It would be nice to have a bit of a framework for unit tests. Currently, there are two such tests (not tagged with "advc.test" but with [advc.enum](#), [advc.fract](#) respectively) executed in `CvXMLLoadUtility::LoadPreMenuGlobals`. There should arguably be a class of tests called PreMenuTest and another called PostMenuTest and perhaps a third class for tests that execute only when a special savegame is loaded.

I've kept a "Giant" world size setting commented out in `Civ4WorldInfos.xml`. I use it for tests with more than 18 civs. It's from vincentz's [34-civ mod](#).

There's a test switch for Permanent Alliances in `UWAIAgent.h`. (They're normally difficult to test through AI Auto Play because the preconditions are so narrow.)

Preprocessor define `MONTE_CARLO_ODDS_TEST` added to `CombatOdds.cpp` (new implementation file, for code moved from `CvGameCoreUtils.cpp`). This causes each calculation of combat odds to be verified through a randomized simulation that closely follows the procedure in `CvUnit::resolveCombat`. I don't know if I've set a sufficient number of samples for those simulations; there might be false positives. Very small inconsistencies between combat odds calculation and combat resolution can't be found this way: It's already very slow with the current sample size; can't increase that by orders of magnitude.

See also	Found bug [001l](#) this way. [003](#): Refactoring changes to combat odds calculations.

advc.repro	Reproducibility test
Rationale	To make sure that the course of a game can be reproduced. Important for debugging, performance tests and synchronicity in network games. It's tedious to pinpoint errors found in the test, but still better than relying entirely on observation and the debugger.
See also	001n: Bufixes for OOS errors.
Config	See description in `ReproTest.h`. The DLL needs to be compiled with the `ENABLE_REPRO_TEST` flag set; otherwise, none of the test code gets executed.
Tbd.	There may still be false positives due to non-synchronized data being stored in savegames.

make	Changes to the DLL compilation process are marked with `advc.make`.

Compiler version: As of v0.96, AdvCiv's GameCoreDLL is compiled using the updated Visual C++ 2003 Toolkit that was included with Visual Studio 2003 SP1 (from 2006). The code can still be compiled with the original 2003 Toolkit (the one linked in the how-to-compile-a-DLL threads on CFC). The updated compiler seems to generate only marginally faster code, if at all.

As for the Windows Platform SDK, a bunch of different versions are available on CFC (see *Credits* below). On Win 8 with VS 2010 Express (and no other version of VS installed), I can use any of these for compiling the AdvCiv DLL and the DLL always ends up with the exact same file size (but not the same contents). Differences in DLL performance either don't exist or are too small to measure them reliably. As with the Toolkit, the files extracted from VS 2003 SP1 seem to be the most recent ones of the proper SDK version (6.0), appear to result in the fastest DLL (for what it's worth; see above) and take up the least amount of disk space. Until AdvCiv 0.98c, I had been using version 7.0A.

Some hints about diagnosing fastdep crashes: `fastdep.exe` dates back to DannyDaemonic's makefile. The VS console will only show "fatal error" when it crashes. For diagnosing the problem, one needs to consult the `depends` file in the `temp_files` folder for the current build target. The last `.cpp` file for which output has been generated was probably handled correctly. The next one in the alphabet should be the culprit. This can be verified by running `fastdep.exe` on that `.cpp` file from a command line; if there's no output, then that file (or I guess an included header) is causing the crash. Beyond that, I think one can only use trial and error, i.e. revert recent changes or delete or comment out code until the crash no longer occurs. So far, the crashes that I've encountered were arguably caused by bugs in fastdep: One time, `#include <stack>` as the last include in `CvMap.cpp` was causing the crash (resolved by moving it to the precompiled `CvGameCoreDLL.h`, which is a better place for standard headers anyway); the other time, an `#if ENABLE_XML_FILE_CACHE` check (cf. 003i) in `CvInfo_Civilization.cpp` (cf. 003x) was the cause; inserting a newline (which looked better too) resolved that issue.

Credits	Got the files from alberts2, who posted them in the Caveman2Cosmos subforum. There appears to be no official download (of just the updated compiler) on Microsoft.com. Caveat: alberts2's download doesn't include some necessary libraries; see list here.
	Windows SDK versions: • v6.0 extracted from VS 2003 SP1 by Anq; this is the one I'm using. Link 1 2 • v6.0 from the "We the People" GitHub account (link), recommended by Nightinggale. • "version 6" hosted here by Asaf. Too recent according to Anq's post linked above. • The install wizard hosted here by Leoreth appears to install v7.0A.
	(Not related to compilation, but I want to document this somewhere: I use a faster version of `Python24.dll` – uploaded here by alberts2. I've seen a speed-up of about 4% in an AI Auto Play test.)

The AdvCiv Makefile is essentially [Nightinggale](#)'s, which is in turn based on [DannyDaemonic](#)'s. (Firaxis only released a `.vcproj` file along with the SDK.) The Visual Studio (VS) project (`.vcxproj`) and solution (`.sln`) files are also based on Nightinggale's.

Added by devolution ([Git commit](#), [forum post](#)): Makefile target `Final_Release` that enables more efficient machine instructions and, crucially, whole-program optimization, which speeds up AI turns by about 10%. (Nightinggale eventually also adopted that – [Git commit](#).)

Tbd.	Test my own changes with VS 2019 (free "Community" edition); only tested with 2010 "Express" so far (available [here](#)). I'm not going to install VS 2019 anytime soon though; too much of a hassle. If I do, [these](#) look like good install instructions. Or maybe there's a nice free version in between 2010 and 2019, one that won't require VS2010 to be installed in addition? VS2013?
	Write a natvis file for VS 2011 onward. [microsoft.com](#) C2C has one, but it might not be easy to port.
	alberts2 is [working on](#) splitting parts of CvGameCoreDLL.dll into a new DLL to be compiled with a modern compiler. If I adopt that change, then VS 2019 will be required. Another idea under consideration by the C2C programmers is a second EXE that will run in a separate process ([link](#)).
	Perhaps replace the makefile with a FASTBuild configuration file, following billw2015's example: Git commit [1](#) [2](#)
	Or otherwise perhaps merge the renamed configurations from Nightinggale's [updated](#) makefile. I should test if this `  Assert|Multi core compilation = Assert|Multi core compilation` looks any nicer in VS than: `Assert-fast|Win32 = Assert-fast|Win32`
	Use the `/GS` option (buffer security check) for debug builds? Since MSVC05, that option is enabled by default. Might slow the program down by a few percentage points. Probably not worth it as accidental security violations seem pretty unlikely.
	C2C is deployed with an archive containing all its dependencies and a script for compilation. If I ever need such a thing, I could follow [this](#) "We the People" commit (which has fewer bells and whistles than the C2C archive, I think); doesn't seem terribly difficult.
See also	[This](#) K-Mod commit may have been intended to enable whole-program optimization, but the `Og` global optimizations option is actually implied by `O2` and is only "global" in the sense that "*the compiler searches entire functions for common subexpressions*" (quoted from the VS2003 documentation).
	BtS had used VCCLCompilerTool properties instead of a makefile: [CvGameCoreDLL.vcproj#L146](#) The `WholeProgramOptimization` property is not used for the final-release build, only `GlobalOptimizations`, which corresponds to the `Og` option.
	There's some more info about the impact of compiler options on function inlining under [advc.inl](#).

Variable `PROJECT_Profile_CFLAGS` added to `Makefile.project` for profiler settings.	

See also	Comments in `Makefile.project`
	Minor tweak in "We the People" that avoids unnecessarily copying `.pdb` files: [Git commit](#) Not merged because it requires a Perl interpreter.

AdvCiv	*Nightinggale's files*

`_NO_DEBUG_HEAP` enabled through the project file.		Reportedly, Visual Studio 2015 and later versions have this enabled by default.
`WX` (treat warnings as errors) added to compiler flags in `Makefile.project`.		I haven't tried it, but I think compilation continues after compiler warnings.
Rationale	Could also make these settings through files that aren't shared with other programmers (`Makefile.settings`, `.vcxproj.user`), but VS 2010 (which benefits from `_NO_DEBUG_HEAP`) remains widely used by Civ 4 modders and, as for `WX`, it makes sense to force all contributors to resolve compiler warnings.	
	`_NO_DEBUG_HEAP` slows down memory (de-)allocation a lot. It also pre-initializes memory allocated on the heap, which can help in finding memory errors, but can also conceal them. Note that the `/RTC1` option (`/RTCs` more specifically) can similarly reveal or obscure memory errors by pre-initializing memory on the stack (local variables).	
See also	The debug heap can be useful for discovering heap corruptions like those discussed under [003k](#).	
	Dynamically allocated memory is pre-initialized in debug builds through overloaded operators; see a few boxes below.	
Obtain the compiler warning level from `Makefile.project`. Set it to `/W4` there. Header `PragmaWarnings.h` added that disables some level-4 warnings that aren't appropriate for this project. The code editor still uses `/W3`. (I guess – it's not explicitly set in the `.vcxproj` file; `/W3` should be the default.) The Windows, Boost and `std` headers are also still compiled with `/W3` (they produce tons of warnings with `/W4`).		`/W3` is set in the makefile.
Rationale	I wasn't sure what warnings I was missing out on, so going to level 4 and working from there seemed like the best approach. About half a dozen of the level-4 warning types turned out to be useful. One could add those to level 3 (which is the approach that Nightinggale seems to be taking for "We the People"; [Git commit](#)), but who knows which other warnings could still turn out to be useful at a later time. I doubt that experimenting with `/Wall` is worth the trouble.	
	I don't think there's an easy way to disable individual warnings in the code editor; `/W4` without exceptions would be too strict.	
Tbd.	I now realize that warnings can also be customized through the `/w` compiler option. Perhaps better to do it that way than through a header file?	
AdvCiv		*BtS*
Instead of 0, set each 32-bit word of dynamically allocated memory to `0xDADADADA` (`-623191334` in decimal). Still only applies to Debug builds.		Dynamically allocated memory is set to 0 through an overloaded `operator new` in `CvGameCoreDLL.cpp`. K-Mod disables that behavior for non-Debug builds.
Credits	Based on [this](#) Caveman2Cosmos revision by billw2015.	
Rationale	To make accidentally uninitialized memory distinguishable from memory that was deliberately set to 0.	
	DADA... is apparently not a widely used [magic debug value](#), but perhaps that's the point – to distinguish it from other allocation functions.	

See also	[003o](#) moves the memory management code to `CvMemoryManager.cpp`.
Tbd.	Perhaps merge improved memory tracking code from Caveman2Cosmos. [Git pull request](#)

AdvCiv	*Nightinggale's files*
Enabled precompiled headers and `jom` for fast `Profile` and `Debug` builds. There are now fast configurations for all targets and slow configurations only for `Debug` and `Release`. I've removed the slow `Assert` config.	From Nightinggale's [thread](#): "*Jom can't build Profile or Debug targets. This is because Microsoft decided that symbols are written to vc70.pdb by the compiler for each cpp file. This means jom tries to have multiple threads write to the same file at the same time. […] I recommend having two Release build rules, one with jom and one without it. The reason is that jom handles files in parallel, but they also write output in parallel. This makes reading error messages tricky […].*" I.e. there are fast `Release` and `Assert` configurations, and only slow configurations for the others.

Credits	billw2015 explains [here](#) how to solve the problem with `vc70.pdb` by using the `/Z7` compiler option.
Rationale	Reading the error messages is very rarely a problem. It's nice to have a non-parallel configuration at hand, but don't need slow versions of each. A high number of configurations makes it slightly more fiddly to switch between configurations.
Tbd.	Should perhaps also throw out the slow `Release` configuration. At least on my system and with VS2010, `/Z7` makes linking far slower than `/Zi`. Not sure if the `/DEBUG:FASTLINK` option introduced by the MSVC05 linker could be used with the EXE (I guess not) and whether it would help much. The non-parallel "debug" configuration should use `/Zi` instead of `/Z7`. I guess a separate build target needs to be defined to accomplish that.

Makefile target `Debug-opt` added for debugging crashes that result from accesses to uninitialized memory. The respective configuration uses `jom` and enables assertions. `Debug-opt` can also be used for inspecting optimized assembly.

Rationale	Even without `_NO_DEBUG_HEAP` and with `/RTC1`, certain memory errors don't occur in a debug build. Optimized assembly is useful for evaluating micro-optimizations.
Config	For debugging, assertions should be enabled by uncommenting the line `#PROJECT_DebugOpt_CFLAGS = /DFASSERT_ENABLE` in Makefile.project. For viewing optimized assembly, it's normally better to keep the assertions disabled.

Use the `/Ob1` compiler option for debug builds. This option allows the compiler to inline functions with an inline keyword. In a test, this decreased the turn time with attached debugger by ca. 30%. Specifically, the MSVC03 documentation says that `/Ob1` "*expands only functions marked as inline or __inline or, in a C++ member function, defined within a class declaration.*" That covers nearly all situations in which `/Ob2` can apply inline expansion, but `/Ob1` seems to do so more reluctantly, i.e. mostly only when a marker is present (and apparently `__forceinline` works too). The difference could also be due to other optimization that have to be enabled along with `/Ob2`.

Config	`PROJECT_Debug_CFLAGS` in `Makefile.project` Can enable further optimizations as needed, e.g. `/Ob2` or even `/O1`.
Rationale	I do most of my testing under the debugger and, toward the late game, turn times do become a bit of hindrance.
See also	[advc.fract](#) and [advc.enum](#) use a lot of indirection. The respective classes hadn't been used as much as now when I did the 30% test mentioned above.

Moved a small part of the CvInfo Python interface from `CyInfoInterface3.cpp` to `CyInfoInterface2.cpp` because the former appears to have been close to exceeding the maximal size allowed for a debug information module.

Credits	[This](#) CFC post by vaxerski suggests to me that the interface file was dangerously large.
See also	[kekm.34](#) splits one of the Python interface files for the same reason.

Flag `_CODE_EDITOR` added to the project file for all builds with assertions in order to make the `FAssertBounds` macro work in the Visual Studio 2010 editor.

See also	Comment in `FAssert.h`

Removed an unused debug flag from `Makefile` and `.vcxproj`: `CVGAMECOREDLL_EXPORTS`

Credits	Adopted from [this](#) Caveman2Cosmos revision by Anq.

Compiler version: As of v0.96, AdvCiv's GameCoreDLL is compiled using the updated Visual C++ 2003 Toolkit that was included with Visual Studio 2003 SP1 (from 2006). The code can still be compiled with the original 2003 Toolkit (the one linked in the how-to-compile-a-DLL threads on CFC). Hopefully, the updated compiler generates slightly faster code.

Since AdvCiv 0.98c, I'm compiling with version 6.0 of the Windows SDK. I had used version 7.0A before without any noticeable problem, but 6.0 seems safer and, based on a single test, results in slightly faster code (at least the test makes me fairly confident that 7.0A isn't faster).

Credits	Got the files from alberts2, who [posted](#) them in the Caveman2Cosmos subforum. There appears to be no official download (of just the updated compiler) on Microsoft.com. Caveat: alberts2's download doesn't include some necessary libraries; see list [here](#). I've obtained the Windows SDK from the "We the People" GitHub account ([link](#)) at Nightinggale's recommendation [here](#).

Meta information from the resource script (`CvGameDLL.rc`) is written into the DLL only when building a final release. K-Mod had already added the mod name (I changed it to AdvCiv) and I've added the mod version under "FileVersion".

Rationale	I don't need the meta info during development. Updating the mod version before each final release seems feasible. Note that `resource.h` is probably only needed for editing the resource script in Visual Studio's resource editor.

Changes to header dependencies. Mostly not marked with any in-line comments.

Credits	Based on Nightinggale's [guide](#) (in spoiler tags under "Fix Firaxis mess").

Rationale	Mainly to reduce compilation time. It's a trade-off: `CvGameCoreDLL.h` gets precompiled (see `PRECOMPILE_CFLAGS1` in Makefile), so if more headers are included there, then compilation becomes faster overall. But a change to any of the headers in `CvGameCoreDLL.h` causes the whole project to be recompiled.
	Caveman2Cosmos moved almost all headers into `CvGameCoreDLL.h`: <u>SVN revision</u> That seems unwise, at least for my purposes, as some headers do change frequently. Also, unnecessary includes obscure the actual dependencies and might pollute namespaces. So I've only moved headers that are unlikely to change and frequently required (or not so frequently required but large).
Tbd.	Perhaps use the <u>/FI</u> (force include) compiler option and remove all the `#include "CvGameCoreDLL.h"` directives. Could argue that it's confusing to have invisibly included headers, but that confusion already exists in the `.h` files, none of which include `CvGameCoreDLL.h` – which works because all the `.cpp` files that include those headers include `CvGameCoreDLL.h` beforehand.
See also	<u>003x</u> splits up `CvInfos.h`. Some of the resulting header files are included in `CvGameCoreDll.h`. (In BtS, the whole `CvInfos.h` was included there.)
	<u>003u</u> cleans up the object-oriented design of the AI classes to an extent, and that has allowed me to include the various AI headers less frequently in implementation files.

AdvCiv	*BtS* (no significant changes in K-Mod/BBAI)
Removed about 20 includes from `CvGameCoreDLL.h` and instead added them only to those cpp files that need them.	`CvGameCoreDLL.h` includes about 50 commonly used header files, some from the Standard Library and Boost, but mostly from the Civ 4 game core. This means that all implementation files need to be recompiled whenever one of the headers in `CvGameCoreDLL.h` changes.
Removed those headers remaining in `CvGameCoreDLL.h` from all implementation files that had included them redundantly.	Many implementation files explicitly include headers that are already included through `CvGameCoreDLL.h`.
Replaced some include directives in headers with forward declarations.	

Rationale	I'm assuming that the headers remaining in `CvGameCoreDLL.h` will remain there permanently.

Wrapper headers `CvGamePlay.h` and `AICore.h` added that include some frequently needed game rule and AI classes.	

Rationale	To reduce the number of include statements.

Reordered the `include` statements in most implementation files a bit so that related headers appear next to each other.	Some patterns are discernible, like including the class's own header file first and EXE-to-DLL interface headers last, but it's mostly haphazard.
As proposed by Nightinggale, I've moved the distance functions from `CvGameCoreUtils.h` to `CvMap.h` and removed the CvMap header from `CvGameCoreUtils.h`. I went a bit farther by making them (non-static) member functions of CvMap – with global wrappers for convenience.	`CvGameCoreUtils.h` contains global utility functions. Some of these deal with distance computations on the game map. Those distance functions operate on CvMap and CvPlot objects and they're inlined for performance, meaning that `CvMap.h` (which includes `CvPlot.h`) has to be included in `CvGameCoreUtils.h` and thus, through `CvGameCoreDLL.h`, in every cpp file.

Rationale	The distance functions don't operate on arbitrary coordinates – they require a CvMap object with specific dimensions for dealing with world-wrap. Tying them to a CvMap object communicates this relationship (which isn't entirely obvious).
See also	advc.pf: Moved the pathfinding functions out of CvGameCoreUtils.

Renamed `BetterBTSAI.h` to `BBAI_Log.h`.

Rationale	For clarity; contains logging code that writes to `BBAI.log`.

Added some declarations to `CvGameCoreDLL.h` to make the Visual Studio Code Editor underline fewer Boost calls.

Credits	Based on "We the People" code by Nightinggale; e.g. from this Git commit.
Rationale	Adding the Boost header paths to Visual Studio shouldn't be difficult to do, but it seems that the (absolute, system-dependent) path will have to be stored in the `.vcxproj` file, which is under source control and should therefore be fully portable.

advc.wine	Debugging support for the Wine compatibility layer
See also	Memory corruption as described under 003k can have worse consequences on Wine than on Windows.

~~All information about failed assertions gets printed to~~ `stdout`. Disabled this again, though the code (not much) is still in `FAssert.cpp` as a comment.	Assertion popups are shown through the `WinUser.h` function `DialogBoxIndirect`, which ~~isn't implemented by~~ works also on Wine.
Information printed to the Visual Studio console is also printed to `stdout`.	`OutputDebugString` in `WinBase.h` is used for writing to the console. These messages aren't visible on Wine.
Rationale	So that errors encountered by Wine users can be diagnosed by providing them with an assert build. I don't know for sure if the BtS debug output is really inaccessible on Wine and why; all based on reports by one (savvy) Wine user; I've never used it myself. It seems that output to `stdout` (via `printf`) is discarded by MSVC applications when no proper console is attached. The format string doesn't seem to get evaluated either, so, considering that none of this happens in release builds and that logging to console is used only sparingly during startup, there should be no need for checking which kind of console is attached; OK to just call both `OutputDebugString` and `printf` in any case.

tsl	"True Starts" map script. The idea is a script that generates a map based on the civs in the game, e.g. with a map portion that resembles Egypt's geography when Egypt is in the game. Not included in the AdvCiv mod so far; only a couple of preparatory changes (see below). Update: I've stopped working on this. I do have a sort of proof of concept, but it would take too much work to complete and polish that. New idea: A game option that causes the most fitting civs to be selected *after* the map and starting locations have been generated. I should be able to reuse parts of the True Starts script.
See also	129: Changes to the standard map generator (resources, features)

AdvCiv	BtS
The Fractal map script may generate land as far north and south as 75-80° latitude. Most map scripts (incl. Fractal) normally don't place any Tundra below a latitude of about 50-55° and the latitude bound for Snow is similarly increased (i.e. moved closer to the poles).	All map scripts with polar caps place bands of water near the poles. For the Fractal script, those bands are so broad that there is normally no land beyond 65-70° latitude. Most other scripts (how?) place land closer to the poles, in particular Pangaea and Continents. On Fractal, Tundra may appear as far south as 44° latitude.

Rationale	44° is about the Alps, way too far south for Tundra. In BtS, this oddity is somewhat evened out by land not extending as far north as it does on Earth. For my TSL script, I'd like to use realistic latitude values, but, regardless of that, I don't think the vertically compressed land has any advantage. On Small map size, the distance between the equator and tundra feels very small; and it's rather more interesting when a passage by ship between a continent and a pole isn't guaranteed to be possible. I think the thick polar water bands may also lead to unnatural (truncated) continent shapes.
Config	I had to hardcode the first part in `CvFractal.cpp` because the code exposing CvFractal to Python isn't part of the SDK. Tundra and Snow placement are set in `CvMapGeneratorUtil.py`.

AdvCiv	BtS
Per-tile latitude values can be set from Python.	The CvPlot class computes latitudes on the fly based on the map dimensions. The standard terrain and feature generators in `CvMapGeneratorUtils.py` have a function `getLatitudeAtPlot`, but overriding that doesn't affect resource placement. To place resources according to custom latitudes, map scripts need to reimplement much of `CvMapGenerator.cpp` (as far as I can tell); PerfectMongoose actually does that.

Tbd.	A map script dependent on a modified CvPlot class isn't nice. Maybe I can find another way later on.
See also	129 tweaks the latitude calculation in the DLL (which applies when the new setter isn't called from Python).

sha	Changes to the merged Show Hidden Attitude Mod (SHAM)
See also	130c: Changes to rank-based modifiers

AdvCiv	SHAM
No display of the low-rank modifier because this has been disabled by change 130c. No option for hiding "spoilers". 130c changes rank-based modifiers so that they don't give away the ranks of unknown civs. Rank-based hate explained as "You're getting ahead of us".	"Developing nations should work together to catch up." "We feel threatened by your large civilization."
First impressions and rank not shown when playing with randomized personalities.	Shows the human first-impression penalty from the difficulty setting – which doesn't hurt but isn't exactly helpful. Hides rank-based modifiers.

See also	[004q](#) hides unusually high memory-based relations modifiers when playing with randomized personalities.
Config	`SHOW_HIDDEN_ATTITUDE` in `GlobalDefines_advc`. Game text in `HiddenAttitudes_CIV4GameText.xml`.
Removed the penalty "This war is going badly for us"; instead "This war spoils our relations" partly based on war success.	-1 "going badly" if their war success is less than ours, "spoils relations" based only on how long the war has been lasting.
Rationale	The -1 doesn't make any difference, a needless complication. The AI not getting as mad if a war sees no action makes more sense. Gameplay-wise, it's mostly a matter of whether (or at which point) a war enemy becomes the worst enemy.
If a non-vassal AI civ would otherwise be Pleased toward a non-vassal war enemy, "... spoils our relations" is increased just enough to make the attitude Cautious.	War enemies can have any attitude toward each other, though anything better than Cautious is rare.
Rationale	Weird if the AI is pleased despite war. Cautious is also weird, but less so, and [UWAI](#) actually increases the willingness for peace a little ("Affection cost") if Cautious despite war. Vassals excluded because they don't choose their wars, and may not participate much; conceivable that attitudes remain non-hostile.

savem	"Savemap" function that saves the map of an ongoing game as a Python script
Credits	Courtesy of xyx ([CFC thread](#)); based on work by tywiggins ([Apolyton thread](#))
See also	See the links above for documentation and possible uses. The description string of the exported map script is generated through DLL code from [106h](#). [127](#) disables the Ctrl+Shift+M AI Auto Play shortcut, which easy to confuse with the key combination that I want to use for savemap.
AdvCiv	xyx's savemap

Can press **Alt+Shift+M** at any point of a game to export the map into a Python script. This shortcut can be disabled on the BUG menu (Map tab).	Needs to be invoked from the Python console.
Will first try to store it in `\AdvCiv\PrivateMaps`. If that location can't be written, `PublicMaps` under `\My Games\Beyond the Sword` is used. AdvCiv will not be able to load it from the latter location unless `NoCustomAssets` is set to 0 in `\AdvCiv\AdvCiv.ini`.	Always save under `My Games\Beyond the Sword\PublicMaps`. I think most mods set `NoCustomAssets=1`, so it's mostly just unmodded BtS that can load map scripts from `\My Games\`.
An on-screen message says whether and where the new file was created.	Feedback only through debug output.
Old files are never overwritten (if I've implemented that part correctly).	Will overwrite if file names clash.
Regardless of the location, Civ 4 needs to be restarted before the exported script can be loaded.	(I don't think a restart can be avoided. In particular, a manual Python reload doesn't help.)
The file name is generated from the name of the original script or scenario, the map dimensions (not if it's a scenario) and initial player count. A number between 1 and 9 is appended if a file with the generated name already exists in the target location. All settings shown on the Settings tab (Victory screen) are included in the description string of the exported scenario. (That string can be viewed in a text editor or through "Play Now".)	The file name can be entered through the Python console. The name of the original script, map dimensions and initial player count are saved in the description string.
Tbd.	Add options for using the original civs and leaders and original difficulty level to the generated scripts. See comments in `savemap.py` about that. For the civs and leaders, DLL functions added by the Change Player mod component will have to be exposed to Python. CvGame could keep a list of tile changes made during the normalization step and then the generated script could reapply those changes depending on whether the original starting sites are used. Probably too much work. As it is, if the original sites aren't used, new sites are computed based on terrain that includes the original normalization changes and no normalization step is performed afterwards.
Can't use savemap in networked multiplayer.	The Python console also isn't normally available in network games.
Rationale	Seems like a pretty convenient way to cheat. In particular, the names of the AI leaders are easy to find in the saved script.
Config	Guard in `savemap.py` at the beginning of the `savemap` function.
When choosing the option to ignore the original starting sites, the selection of starting sites is left entirely to the DLL.	New starting locations are generated in Python through `CvMapGeneratorUtil.findStartingPlot`.
Rationale	So that StartingPositionIteration (change [027](#)) is used.

advc.rh	Modular XML loading bugfix merged from rheinig's mod ([CFC link](#))

Credits	xyx made me aware ([here](#)) of this fairly obscure mod.

From the mod's readme file:

"*Some XML tables are loaded in two passes, and the code did load the second pass data into the wrong records for modular additions. Ever wondered why the tech tree or promotion prerequisites get mucked up if you modularize them? Now you *can* modularize those. The full list of Infos that define a second pass loader method and the values those load is:*

CvTechInfo(OrPreReqs,AndPreReqs), CvPromotionInfo(PromotionPrereq,PromotionPrereqOr1,PromotionPrereqOr2), CvCivilizationInfo(DerivativeCiv), CvImprovementInfo(ImprovementPillage,ImprovementUpgrade), CvProjectInfo(PrereqProjects,AnyonePrereqProject), CvEventInfo(AdditionalEvents,EventTimes,ClearEvents).

Note Buildings and Units don't need to do that as their perceived self-referencing prerequisites actually refer to the respective "Class", and those are loaded far ahead. Otherwise, this bug may well not have slipped by QA.

Update - Version 3.1.7.3 now delays pass 2 of the monolithic XML until after pass 1 of the modules has finished, too. This finally fixes modular random events.

Changed modular XML load order:

Fixed CvXMLLoadUtility::LoadGlobalClassInfo to sort the list of files after enumerating them and before loading them. Why? Well, before the load order was determined by the file system and essentially unpredictable. You could easily be working on non-XML files in those directories and suddenly have units, buildings or whatever swap their IDs, thus having saved game incompatibilities the loader couldn't catch. The sort now is case-sensitive, which is bad, but I'll leave it at that as the rules governing the casing of returned names are deterministic and constant over any OS generation. You might still get into the effects I mentioned if you moved from, say, Windoze 9x (urgh!) to NT (sorry, I mean XP), but does cIV even run on 9x???"

Rationale	For mod-mods that use modular loading. The issue doesn't sound important, but, despite the lengthy comments, the code changes are very minor.
Not merged	"*Major Enhancement: Incremental modular loading*" — That's a bit more work to merge and even more work to test as I don't know how modular loading is exactly supposed to work. For what it's worth, it seems preferable to the "XML COPY" code by MRGENIE in RoM-AND2. Most of the non-XML bugfixes in the mod seem to have been included in the unofficial patches. The rest I seem to have fixed myself (some 10 years later). The minor optional rule changes and changes to help text, ultimately, aren't worth including in AdvCiv.
See also	This last item of the readme file has inspired a similar change tagged with "advc.006": "*CyGlobalContext().getInfoTypeForString, when called with an unknown key string, formerly logged an error in xml.log citing the last xml file loaded even if the call came from Python and had nothing to do with XML.*"

kekm	Bugfixes (and other changes) from DarkLunaPhantom's Kek-Mod [I had labeled these with "dlph" at first because Kek-Mod didn't have a name yet.]
Credits	[Source](#) (Kek-Mod) I've only adopted those changes that were easy to make (cost/ benefit), and none that only deal with PBEM or Pitboss.
Tbd.	The PBEM bugfixes would be nice to have too. Even if I don't test them; I doubt that they's break something. (The Pitboss changes might be too much work.)
See also	[250c](#) also includes a couple of changes from Kek-Mod [001](#): My own misc. bugfixes I've reported some minor issues with Kek-Mod changes [here](#).
kekm.1	"*Circumnavigation bonus is now always preserved when forming permanent alliance, previously it was only when the player having it had smaller team number.*" (also fixed in K-Mod 1.45)

kekm.2	*"Fixed a (...) bug which caused unremovable espionage city visibility after forming permanent alliance for all cities that the player with larger team number had visibility of at that moment."*
	(As far as I understand, only kekm.26 fixes this properly.)

kekm.3	Defensive pacts despite war	

AdvCiv		*BtS*
BBAI option for defensive pacts despite war enabled, and adopted a bugfix and extension from Kek-Mod: *"now enable[s] defensive pacts to be signed while at war."* But the AI only signs a DP when sharing all wars; cancels DP when wars are no longer shared. And I'm allowing defensive pacts to be canceled (0 turns to cancel) after a DoW.		When war is declared on a civ, the defensive pacts of that civ take effect and are then canceled. BBAI has an optional rule change that leaves defensive pacts in place after taking effect; disabled by default. A civ that declares war loses all its defensive pacts. Can't sign defensive pacts while at war.
Rationale	May not make defensive pacts a lot more useful, but it's more plausible this way. As for my adjustment: "We'll aid you against any further aggression, but, in your current wars, you're on your own" doesn't sound like a typical military pact.	
	The immediate cancelation when no longer sharing all wars happens in `CvPlayerAI::checkCancel`. If that code is removed, then 133 will handle the cancelation, resulting in a probabilistic delay. This could give the DP ally enough time to make peace, preserving the DP. After a test, I think a DP will rarely survive a peace deal despite the delay, and it's cleaner to cancel the DP immediately.	
See also	130y reduces the diplo penalty from DoW triggered by a DP. 104i makes the AI willing to talk with all DP allies upon making peace with one of them. kekm.25 allows DP votes while at war.	
Tbd.	What happens when A has a DP with B and C has a DP with D and A declares war on C? I suppose they'll all be at war (that's how it works in Dawn of Civilization too); is that desirable?	

kekm.4	Can't gift cargo if it contains units that can't be gifted.
See also	An addition to 123a
kekm.5	*"Obsolete buildings and unused power plants (e.g. Nuclear Plant without Uranium or in a city that also has Hydro Plant or receives power from Three Gorges Dam) cannot trigger meltdown event anymore."*
See also	652: Other rule changes to meltdowns.
kekm.6	Barbarians can't build spies.
See also	307 prevents the Barbarians from training some later-era units.
kekm.7	Can nuke despite neutral units; they take no damage.
Rationale	If the neutral units took damage (without having to declare war), this would again be exploitable. Units in nuke range not taking damage is counterintuitive, but I don't see a better simple solution. In the future, I'd like to restrict the effect of nukes to a single tile; then it won't be so strange anymore that only hostile units are hit because one can imagine that units of different owners are encamped separately.

kekm.8	"*Fixed bug in AI evaluation of gifted unit for the purpose of relations bonus.*"
Not merged	"*Player cannot gift combat units to third party which in war with rival with whom the player has unbreakable (temporary or permanent) peace treaty. Also, the receiving player now must satisfy technology requirements (both for the unit itself and for the prerequisite resources).*" [link?]
	"*Free units from tribal villages cannot move in their first turn. Gifted units are immobile only if they change teams.*" Git commit <u>1</u> <u>2</u>
	These aren't restrictions that players would intuitively expect, and I'm not sure that they're necessary. (I do think that there should be diplomatic consequences for gifting units beyond "traded with our worst enemy.")
kekm.9	"*Fixed how AI and worker actions treat fallout feature. Fallout had bNoImprovement set to 1 signaling to AI that no improvements can be built there which didn't make sense since fallout can be cleared just like forest and jungle can. This made AI far less likely to scrub fallout (if at all) than to chop jungle although the features are similar. Improvements can now be built directly on top of fallout and fallout will be cleared in the process just like it is with forest and jungle.*
	Fixed interface and AI bug that caused the game to sometimes wrongly show an improvement as unbuildable. Sometimes the game doesn't [realize] *that an improvement can be built. E.g. plains jungle yields 0 food so farm cannot be built as it requires 1 food, but farm can be built there by chopping the jungle as the underlying plains terrain yields 1 food. AI followed the same slightly flawed logic as did the interface. My fix for fallout feature introduced many more such situations.*"
kekm.10	"~~Added verification of state religion. Player shouldn't be able to keep their state religion when there are no cities with it anymore.~~"
Rationale	Disabled this again. I don't want to force a civ out of its religion while it may still try to reconquer its holy city.
Config	Disabled through `CvPlayer::doTurn`.
kekm.11	"*Fixed advanced start bug with reduced unit costs. Advanced start code is bad in general so I just fixed the immediate problem. (There was a way to get extra gold by refunding a worker while having expansive trait.)*"
kekm.12	"*Removed Disorganized promotion from free barbarians boats with hidden nationality. Hidden nationality units shouldn't be too obviously (non-)barbarian.*"
	(But I'm not allowing Barbarians to have Privateers.)
kekm.13	"*Fixed permanent alliance bug with AP/UN leader. Similar as with other permanent alliance bugs, if AP/UN leader had higher team number than its permanent ally an empty team would become AP/UN leader instead of the newly formed team.*"
Not merged	"*Changed condition when is a team considered AP/UN full member (and thus eligible for election). Previously all team members had to be full members, but now at least one is enough.*"
	I don't have an opinion on how this should work.
kekm.14	"*Set barbarian unit* [gold] *costs to 0. These costs don't affect barbarians directly, but they still do influence some decisions.*"
	Git [commit](#)

kekm.15	"*Moved the code for building missiles for missile carriers outside of assault only case. K-Mod made similar change for building planes for carriers already so I did the same for missiles. Also fixed a bug when checking whether there are enough missiles already.*" Git [commit](#)
	"*Fixed a bug (?) in missile production AI. Missiles for filling up missile carriers should be built in low productivity cities and not high productivity cities.*" Git [commit](#)
kekm.16	"*Bomb Shelter was previously always evaluated as almost completely worthless (!!!). I think that AI should build it as soon as possible pretty much everywhere as it is very cheap and effective.*"
	Git [commit](#)
Not merged	"*SDI evaluation previously ignored that nukes can quickly obliterate entire militaries and there were some issues with estimating number of nuclear attacks.*"
	Too much work to merge considering that the rules for nukes will have to change at some point, and that the AI code will have to be revised afterwards.
kekm.17	"*Game era calculation changed from rounding down to rounding of all players' average era.*"
	Git [commit](#)
Rationale	I had always assumed that it worked this way, and I'm frequently referencing the game era in AI and Barbarian code, so this should be a significant improvement.
Not merged	"*Added Advanced Settlers game option. When used, settlers behave similarly like in Advanced Era starts, i.e. new cities start with more buildings and population as the game progresses through eras. Settler era is a minimum of player's era and game era [...]*"
	I want to add something like this, but I'm not quite happy with the implementation. Looks like it could reward stockpiling Settlers, and the cost of a Settler can increase while in production if the player or game era changes, which is a bit messy. Alternative solutions implemented in other mods: A second Settler unit ("Colonist") with a tech requirement in late Renaissance (which mod was that?) and Realism Invictus's Ministries. Another idea: A production bonus for Ancient and Classical buildings from e.g. the Steel technology.
kekm.18	"*Hidden game options are now always set to their default value as they cannot be changed or interacted with anyway and switching between mods can mess up those options.*"
	(But I've implemented it through `CvInitCore::resetGame`.)
kekm.19	"*Capital cannot be moved while spaceship is underway. Reason is the fact that capturing the capital destroys the traveling spaceship, so capital cannot be hidden in this way after the launch.*"
Tbd.	Would prefer to let the production cost of Palace increase over the course of a game. This could also make it easier to move the capital in the early game when starting in an awkward spot.
See also	[cdtw.6](#) causes the AI to move its capital when close to a Space victory.

kekm.20	"*Fixed Bomb Shelter effect for non-combat units. Bomb Shelter used to reduce the probability of nuke destroying a non-combat unit from around 80% to around 2-3% (these probabilities are not immediately obvious and have to be calculated; exercise is left for the reader) because someone was not careful with probabilities.* `NUKE_NON_COMBAT_DEATH_THRESHOLD` *was probably picked so that the probability is close to average nuke damage to combat units (which is 79%).*
	I changed the chance to destroy a non-combat unit to exactly 79%, and Bomb Shelter halves that so those are now exactly the same as average damage to combat units. `NUKE_NON_COMBAT_DEATH_THRESHOLD` *is now unused.*"
	Git commit, later bugfix
	The formula sounds legit; just copied it.
See also	advc.650 changes the damage formula for combat units, but I think everything in DarkLunaPhantom's comment still applies.
kekm.21	"*Show barbarian territory on the minimap and in the globe view.*"
	Git commit
kekm.22	"*Changed average handicap* [in multiplayer] *from round down to round.*"
	Git commit
See also	250a bases that computation on difficulty values assigned to each handicap through XML.

kekm.23	*AdvCiv*	*Kek-Mod*
	Razing: Not merged; no tile culture is removed. Culture after trade is handled by change ctr, which converts only a fraction of city and tile culture. I've adopted the equal treatment of liberation and regular city trade from Kek-Mod. I've merged this change. Cities liberated upon the creation of a colonial vassal still receive the free units; that's also the case in Kek-Mod.	"*Reworked how culture behaves when a city is razed or acquired. Razing a city will now erase (most of) its plot culture. Trading a city will transfer both city and (most of) plot culture to the new owner. Not all plot culture will be included because plot culture is generated by different mechanisms (e.g. trade routes and free plot culture) and increasing and decreasing city culture doesn't result in adding and removing the same amount of plot culture. Also, liberated cities don't get free units just because a vassal acquired them and diplomatic votes resulting in changing of city ownership will leave old owner's culture intact. An attempt to get more consistent and sane plot/city culture effects related to city trades.*" Git commit
Rationale	A single movement point spent on razing a city shouldn't be enough to ethnically (let alone culturally) "cleanse" a region. Likewise, city ownership agreed to on paper shouldn't instantly resolve all ethnic/ cultural conflicts in favor of the new owner.	
Tbd.	Razing should cause some loss of tile culture.	

kekm.24	"*Colonies don't inherit espionage points from parent civ anymore (this didn't seem fair or necessary), but they do inherit EspionagePointsEver.* *Colonies cannot reuse player slots anymore as this can cause weird bugs sometimes and this is the easiest fix. Number of players can be easily increased (it already is 48) so this is not necessary.*" [Git commit](#) I'm still allowing human players to reuse a slot when creating a colony.
Rationale	The maximal civ count in AdvCiv is only 18 and increasing it comes with a performance penalty. Reusing slots is not really compatible with the [Immortal Culture](#) changes (I suppose culture will be set to 0 when the colonial vassal is initialized), but I don't want to disallow colonial vassals entirely in games with 18 civs. If the player doesn't like the result, he or she can hopefully reload a savegame. (Whereas the player would be unable to prevent the AI from reusing a slot – if that were allowed.)
Tbd.	Different approach: Store tile culture in an `ArrayEnumMap<CivilizationTypes,int>`. That'll add some 15*4 byte to CvPlot and may also slow down some algorithms. Peanuts though compared with raising the civ count. Upd.: But the same civ can be played by several players at the same time, so it won't work quite this way.
See also	[001](#): A couple of bugfixes pertaining to colonial vassals and the Random Personalities option.
kekm.25	"*Reworked resolutions. Fixed multiple bugs and inconsistencies with conditions for proposing and defying resolutions. [...] Vassals cannot defy resolutions anymore. Players can defy resolution assigning them a city. AI can now choose (and vote) to repeal resolutions.*" Git [commit](#) Changed it so that only capitulated vassals are unable to defy. Voluntary vassals can defy everything except war and peace votes (in BtS, they can defy everything, in BBAI they can defy peace votes but not war votes). Regarding a comment in `CvPlayerAI::AI_diploVote` about vassals and friends of the Secretary General in repeal votes: "[O]*nly important if the Secretary General plans to vote yes. Incorporating the other case properly would be a lot of additional work....*". I've tried to do that extra work; hope I got it right (so that friends and vassals help the Secretary General repeal resolutions). "*Human vassals are forced to vote for their master. Unless they are also a candidate themselves.*" Git [commit](#) Only relevant for the [R&F](#) option because humans can't normally become vassals in AdvCiv. I'm only applying this to capitulated human vassals. "*War resolution can now be proposed against voting members (and non-members). Defensive pacts with attackers are cancelled before implementing war resolution.*" I don't think this ever works the way it's implemented in Kek-Mod. `CvGame::canDoResolution` checks if all voting members – including the target! – are able to declare war on the target. I've corrected that: Now only full members are checked, only they declare war and only they can defy the resolution. These change are tagged with "kekm.25/advc". Moreover, only peace treaties of the AP leader can block a war resolution; peace treaties of other members don't matter. (But the AI tries to honor its promises by voting against war when there is a peace treaty.)

Rationale	Shouldn't expect human players to keep track of peace treaties between AI civs, and don't want to allow players to block war votes by signing peace treaties (e.g. by asking for a gift). K-Mod already allowed vassal agreements to override peace treaties.
See also	130f deals with stop-trading resolutions in a similar way. CFC thread about a BtS bug fixed by the first commit listed above.
Tbd.	Looks like a human war vote target (non-full member) will now get to vote. Should be an automatic vote against instead. That's handled by `CvGame::addVoteTriggered`. Planned changes to membership rules (see 178) will address the problems with war votes in a simpler way. And I don't think I want civs without a state religion to be eligible targets for war votes.
Not merged	*"Religious population for votes in AP is now divided by number of religions in the city. Defiance penalty is now given to the whole team if one team member defies a passing resolution."* This penalizes civs too much for heaving multiple religions and might weaponize Missionaries of non-AP religions. Also, a state religion tends to have larger communities than a non-state religion (and a civ with a state religion not matching the AP religion already has its votes reduced). Not sure about the defiance penalty. If civs vote individually, then it seems that they should also bear the consequences individually.
See also	130v makes capitulated vassals vote along with their master and places many other restrictions on capitulated vassals. 178: My own changes to AP votes
kekm.26	Git commit "*Finally fixed the bug with espionage visibility. Adjusted CvTeam::shareItems so that only use is enough.*" (Cf. kekm.2) "*Scaled third party counter towards new team with number of players in each team for consistency.* *Fixed bug with no tech brokering status not being preserved properly.* *Espionage points of the new team and towards the new team are now sum of the old ones instead of max.*" All this only concerns Permanent Alliances. "*Changed how multiple war declarations work. declareWar used to nest war declarations, now they are queued to trigger defensive pacts and everything else in the correct order.*" This may also be an improvement in other situations; hard to say. There was a bug preventing updates of the attitude cache; fixed. The bugfix was also included in Kek-Mod 0.25.

Not merged (for now)	"*Increased victory conditions for permanent alliances. Number of required spaceship parts is multiplied by (1+0.5*(number_of_players-1)) rounded up. Number of required cities for cultural victory is multiplied by (1+0.5*(number_of_players-1)) rounded down.*" Git commit
	"*Adjusted domination thresholds for permanent alliances. Minimum land percentage is now 100*(1+n)/(3+n)+1 where n is number of players in the team. This used to be 51 for all teams. For domination population, in addition to 25% lead required, the population of the second best team is scaled with the standard ratio of (1+0.5*(n-1)) values for both teams (with a minimum of 1), i.e. larger lead is required in the case that the second best team has less players.*" Git commit 1 2
	Perhaps very sensible changes, but it's difficult to say without really having played with Permanent Alliances. This would have to be mentioned somewhat prominently in the main part of the manual.
Not merged	"*Changed how maximum distance is calculated*"
	See 140
Not merged	"*Always show Dawn of Man when beginning a new game*" Git commit
	Already handled by 704, 250c.
kekm.27	"*Added OOSLogger. OOSLogger creates a file with all the data used for computing the sync checksum when OOS is detected. Idea from Fall from Heaven 2 by Kael. Implemented as a slight modification of the version in ExtraModMod by Terkhen. Implemented as a BUG module.*"
	Git commit 1 2
	The log is created in the Logs folder and named "OOSLog". I've added a MessageLog check (to make sure that both players are aware of the logging), fixed a few of bugs and made some minor adjustments. These were also adopted by Kek-Mod 0.25.
Not merged	"*Improved OOSLogger*" Git commit Not really an improvement for my purposes.
Tbd.	The log is sometimes triggered because game scores don't match. On the next time slice, scores are updated by CvGame and back in-sync. Not sure if this is a false positive or a problem with `CvGame::setScoreDirty` somehow getting called asynchronously.
	Move this mostly into the DLL? A CvGame function would be easier to keep updated with the OOS checksum computed by CvGame. And in Python, syntax errors go unnoticed and then parts of the log is lost when an OOS error actually occurs.
	To be merged: A similar but hopefully more potent tool by Nightinggale: Git commit
	Logging of game net messages by alberts2: SourceForge revision
kekm.28	"*Barbarians also get some starting tech in advanced start. […] Techs they get in advanced start are the average of all player's tech status after advanced start.*" Git commit
	"*[...] [D]on't try to give free units or advanced start points to barbarians.*" Git commit
See also	The rest of that second commit is covered by 250c.
Not merged	"*Barbarians require revealing and enabling tech for resources to be able to get units which require those resources.*" Covered by 301.

kekm.29	"*Fixed a bug in target city evaluation for cities that would be autorazed.*" Git [commit](#) Obsolete; fixed differently now.
kekm.30	"*Added leader and civ icons to scoreboard.*" Git [commit](#) Disabled by default.
Not merged	"Unmet dead civs can now be shown in scoreboard." Git [commit](#) "*Name, civilization and color of dead players are not concealed.*" Git [commit](#) Covered by [004v](#).
kekm.31	"*No tech brokering for techs acquired by Internet.*" Git [commit](#)
Not merged	"*Show a message when a player abandons its state religion.*" Git [commit](#) Covered by [150a](#).
kekm.32	"*Added choice of map wraps to Not to Big or Small mapscript.*" Git [commit](#) Indeed strange that the map didn't have that option. A caveat: World-wrap for MultilayeredFractal-based maps is implemented through `iTerrainFlags`, so these flags mustn't be removed from `generatePlotsInRegion` calls.
See also	[advc.mxc](#) adds a world-wrap option to the "Continents and Islands" script.
Not merged	"*[O]ption to adjust water percent in not_too_Big_or_Small. Default water percent in not_too_Big_or_Small is 74 and sealevel options can change that by -8/0/+6. Adjust water percent option can change that by 0/-5/-10/-15/-20/-25.*" Git [commit](#) Too extravagant. "*Added a new climate [...] called Warm and is mainly characterized by less ice and tundra. It is available for all mapscripts that use default climate system. Created by AjmoCiv.*" Git [commit](#) Not as evocative as the other climate settings.
Not merged	"*Added Gigantic map size. It is larger than Huge. A lot of values in CIV4WorldInfo.xml seem arbitrary, but I tried to preserve some kind of pattern.*" Covered by [advc.test](#).
kekm.33	"*Changed espionage costs for teams. I want costs to scale with 1+0.5(number of members - 1), but since there are two teams (and two directions) involved, it will scale with the square root of the ratio of those values. Idea for formula by Fran.*" Git [commit](#) (Fran must be a player at "Zulan's Civilization [corner](#)") In BtS, mission costs aren't adjusted to the team size at all. K-Mod 1.45 multiplies the cost by the team size. The same handicap (+50% per team member) should apply as for tech costs – which is what the Kek-Mod formula does.
Not merged	"*Changed how unit type for draft is determined. [...]*" Git commit [1](#) [2](#) [3](#) [4](#) [5](#) [6](#) [7](#) [8](#) Too much work for too little gain.
kekm.34	"*Split CyCityInterface in two parts. Its size was obstructing compilation of debug DLL. Some minor spelling corrections in changed files.*" Git [commit](#) Large Python interface files can lead to a "debug information module size exceeded" error when compiling a Debug DLL. I've never encountered that error, but keldath has confirmed it. I guess it's code bloat caused by the `boost::python` macros.
See also	According to Nightinggale ([CFC post](#)), the file size limit is imposed by Boost. (So I guess it applies throughout the GameCore codebase since Boost headers are included in the precompiled header.)

kekm.35	"*Changed some details of how starting locations are picked. [...]*" Git commit 1 2 3
	"*Starting area picking system is changed so that it doesn't overvalue large, but very bad areas. When picking starting location, locations with very little food (before normalization) will be avoided if possible to avoid starting on the edge of very bad terrain (after normalization).*" Due to 027, the choice of the starting areas normally only matters as an initial solution for my own starting position algorithm. Still, a better initial solution is better, i.e. may lead to a better final solution or may at least save time.
	"*Fixed a bug in location evaluation in case when starting location needs to be picked again in advanced start.*"
	"*In advanced start, the possibility of automatically exchanging starting location with a teammember under certain conditions is now removed. (Why was that even there?)*" Merged but **disabled**. It seems that the BtS code doesn't have any adverse effect and perhaps it does somehow lead to a fairer turn order in team games.
	"*Strengthened starting location (pre-normalization) food prerequisite.* *Instead of 5x5 area, the whole standard advanced start sized area is checked for food.*"
Not merged	"*Starting location picking system is now the same in every game mode. After randomly assigning preplaced starting locations, it first picks locations for some number of AIs depending on handicap, then for all humans and then for the rest of AIs. Order doesn't depend on player id numbers. This is done this way because locations picked earlier are usually better.*" Covered by 108b except perhaps for team games, but all that is a bit complicated, so I'm not going to bother with it (again).
Not merged	"*Require debug mode for cheat actions and tooltips.*" Git commit Covered by 135c.
kekm.36	"*Wonder list now uses player text color instead of primary color.*" Git commit "*[A]dded calculation for number of non-early religions, [...] primarily for mod compatibility.*" Git commit
Not merged	"*Adjusted era factors in calculations [...], primarily for mod compatibility*" Git commit Addressed by advc.erai.
kekm.37	"*Fixed a crash when pillaging is intercepted by sea patrol. [...]*" Git commit I don't think I ever experienced that crash, but it sounds plausible that it could happen.

kekm.38	"*Internet now counts players and not teams. […] Internet's description and tooltip mention civilizations, but the code used teams. I changed the code to agree with the description because that seemed to make more sense for permanent alliances. Additionally, each aditional team member now increases the civilization count requirement by 1.*" Git commit
	The last part doesn't make sense to me, so I haven't adopted that. The alliance already takes away one player from whom tech could spread via the Internet, so why increase the threshold for tech spread in addition? And, depending on the total player count, a threshold of 3 can be a lot more strict than 2.
	As for balance in proper team games, in BtS, the Internet is less powerful in team games than in non-team games, and the Kek-Mod change reverses this – because, in team games, there is less entropy in the distribution of techs among players. E.g. in a game with 4 teams of 2, the Internet will grant all techs that 1 of the 3 other teams knows. I woud prefer to make the Internet just as strong in team games as in non-team games, but there's no middle ground to be had. In particular, increasing the threshold from 2 to 3 in games with teams of 2 would be no different from increasing it to 4 or counting teams instead of players (as does BtS).

kekm.39	Fixed int overflow in `CvTeamAI::AI_endWarVal` Git commit (Less of an issue in AdvCiv because UWAI bypasses that function.)
	Great Mediator event: "*Added a check to avoid weird bugs like making peace with a vassal if the situation changes after the event is triggered.*" Git commit

k146	K-Mod update 1.46
	"*Fixed estimateCollateralWeight with non-combat units.*" "*Fixed potential divide-by-zero in AI_estimateBreakEvenGoldPercent*" "*Fixed overflow bug in CvCity::doPlotCultureTimes100*" Not merged; AdvCiv had already fixed these. "Adjusted of attitude of human players toward AI." Not merged; I think I got this covered already. "Restored WHEOOHRN scoreboard indicator." Not merged; see 210a. "*Tweaks to default options.*" Not merged: "*Great person bars should be on by default.*" (Disagree. Perhaps for people with wide screens because the bars fit on a single row then, but not everyone has such a screen.) "*Increased trade culture rate. Instead of 1% per culture level, it's now the average of that and the max rate. ie. percentage = (current level + max level)/2;*" Merged although trade culture is disabled by default in AdvCiv (see 125). "*Updated loading hints: [...] added [...] alt-wake [...]*" That's the only new hint I've merged (press Alt to wake up units worldwide). "*changes to AI_techValue [...] Increased chop value.*" I had to water this down a bit because, in combination with change 036, the AI was prioritizing Bronze Working and Iron Working too much. The remaining changes, i.e. the bulk of the update, are in the back end of the AI and I've merged them almost without adaptation: "*Fixed handling of AI_follow actions which split the group.* *CvSelectionGroupAI::AI_update no longer assumes that the group stays intact with CvUnitAI::AI_follow actions.*" This fixes a fairly rare non-reproducible crash-to-desktop bug.

"*Added CvPlayer::haveResourcesToTrain*
This is used to help with AI decisions when it isn't clear which city we'll be building in."
Used in just one place so far though.

"*Decreased food devalue rate. (Fast growth is more highly valued.)*
bEmphasizeFood causes more optimistic evaluation of slave-whipping.
SlaveryValue re-written to use a 'devalue rate' system.
growthValue now takes into account the food cost of worked jobs.
'plotMagicValue' now (usually) assumes fully-upgraded improvements.
'Emphasise food' should work better now.
CvPlot::getYieldWithBuild bWithUp now uses full upgrade rather than 2 stages."

"*AI_updateCommerceWeights now better understands focused espionage.*
Having very high values on the espionage weight slider are now taken to mean that you don't want to spend espionage on the other teams. This reduces the chance of the AI choosing to assign spies inappropriately. Previously, having a non-zero weight against a team was taken to mean that the player wanted to [have?] *espionage points against that team - and thus espionage was more highly valued if the points were low. Now that is only the case if the weight is above a certain threshold. The threshold is based on total weight, and number of teams etc.*"

"*Minor tweaks to CvCityAI::AI_buildingValue*
Reduced building value of free-tech. (The evaluation for this is still just a very poor guess; but it is very difficult to evaluate it properly. A true evaluation would require details which we currently do not take into account.)
Reduced the value of espionage commerce multipliers."

"*Many changes to AI_techUnitValue*
We now use slightly more detailed war state info: bLandWar, bIsAnyAssault.
Default AI type max value weight increased from 100 to 250, but calculated weight decreased. (ie. Units that are good at their default role will now be valued much more highly; but units that are poor will be valued slightly lower.)
Increased max value weight for non-default types (from 100 to 150).
Adjusted most military values based on bLandwar status, as well as other strategy adjustments; such as for ECONOMY_FOCUS.
Added an optimistic value bonus for tech which reveal required resources."

"*For UNITAI_COUNTER, reduced value of attack modifier, added value for defence modifier. Reduced value of speed. For UNITAI_CITY_DEFENSE, added value for hills defence.*"

"*Tweaked small-city yield evaluation. Yield evaluation now assumes that cities with population < 3 will want to grow, even if there are currently no good jobs. This helps prevent flip-flopping with food plots.*"

"*Many changes to AI_techValue.*
Disabled the random value bonuses for techs with iPathLength > 1. (Raw bonus, and bonus from wonders). ie. techs for which not all prereqs have been met no longer get these random bonuses.
Note: This is partially to help the AI focus on real benefits; but a big reason for removing this randomness is a technical one. [...] *changing the way multi-step research paths are evaluated. Having randomness in prereqs could upset the evaluations because the values will be used for several techs.*
Decreased random wonder value.
Increased the value of gold trading.
Changed several constant values to scale with the number of cities. (These things don't necessarily have anything to do with the number of cities. We're only multiplying by cities so that the value scales like things which are evaluated properly.)
Moved project evaluations to a new function: AI_techProjectValue.
Increased value of civic improvements.
Decreased the value of religions.
Rewrote evaluation of bonus techs. It's now based on the average of the max tech cost and average tech cost for currently researchable techs.
Renamed bIgnoreCost to bFreeTech, to better reflect what the argument is used for. Rescaled the return value so that it is roughly comparable for free and non-free tech.
Renamed "tech whoring" to "tech ground-breaking", to better reflect the concept. (ie. getting the tech first, for better trade options.) Slightly increased the value."

"*Rewrote CvPlayerAI::AI_bestTech to better consider tech pathways.*

Previously the AI would choose to beeline high-value techs within the depth limit without considering the prerequisites at all. This sometimes led to poor choices; eg. getting stuck on very high cost prereqs, or researching prereqs which would obsolete important bonuses. In the new system, the AI only ever chooses techs that they can research; but adds value based on which techs follow on.
NOTE: this is a totally new system which will require some changes and balance."
I had fixed these two bugs and another one mentioned here by Mattygerst, but replaced my bugfixes with the code karadoc released in early May 2018. I've decreased the `DepthRate` so that the AI is less interested in beelining toward powerful techs. I worry that a high `DepthRate` makes AI tech paths too predictable.

"*Minor changes to AI unit movements.*
Tweaked attack threshold for AI_attackCityMove. Increased the acceptable number of ship loading turns AI_attackCityMove
Disabled the 'smart' danger aversion in AI_pirateBlockade. (It was computationally expensive, and not particularly effective.)"
I've written a cheaper danger check as a replacement.

kmodx	Bugfixes from K-Mod Extended
Credits	By alberts2 **Git repository** Much of it just improves hazardous code, but there are also actual bugfixes, e.g. comparisons between different enum types and a mix-up of inner- and outer-loop variables. Hard to say how significant these bugs were, but good riddance! I'm listing the descriptions of the merged Git commits below. K-Mod 1.45 includes these fixes as well; I had merged them separately before v1.45 was published.
	"*Fixed a MemoryLeak in CvDllPythonEvents::reportSelectionGroupPushMission* *Fixed a memory leak in CvGameTextMgr::setCombatPlotHelp* *Fixed uninitialized variables in CvInfoWater.cpp* *Fixed the CyGameTextMgr(CvGameTextMgr* pGameTextMgr) constructor* *Fixed a few coding errors* *Fixed various coding errors* *Modified BUG's WidgetUtil.py to make it compatible with the traditional means of specifying custom Python widgets via WIDGET_PYTHON*"
See also	I've run the code through Cppcheck, went through about 1000 (style) warnings and fixed numerous minor oversights. My corrections are mostly unmarked (as the point of the changes is often to improve readability, and tags in the code would run counter to that), or else marked with id 003 (or 001 in the case of minor bugfixes). Several hundred warnings remain, which are more or less false positives as far as I can tell. 003j deals with unused functions discovered through Cppcheck specifically.

cdtw	AI changes by Dave_uk
Credits	Source (LoR SDK ModMod) Dave's changes in the LoR code are tagged with "CD Tweaks". I'm not sure what the "CD" stands for – collateral damage? – but, anyway, hence my tag "cdtw". I've adopted only a small portion of his changes because some are only important for LoR (with its focus on modern warfare), many superseded by K-Mod/AdvCiv and a few I didn't find worth having (e.g. because of a planned gameplay change or too minor). And I've made some small changes to the code I did adopt.

cdtw.1	"*vassals are more keen to keep* [rather than raze] *cities, as they can't declare war themselves to expand*"
cdtw.2	Regarding the AI choice of target city when in a Blitz or Fast-mover strategy: "*when blitzing place higher value on cities with no defense modifiers*"
See also	[104d](#) always takes into account tile defense when choosing a target city.
cdtw.3	AI evaluation of tech that unlocks a process (Alphabet, Currency, Drama): "*value good processes if we already have lots of units, and are not at war with anyone*".
	Disabled this again after some testing. Early unit spending doesn't seem to be an issue in K-Mod/AdvCiv.
cdtw.4	Greater willingness to switch to Police State when war weariness is painful.
cdtw.5	"*if we're worried about being attacked, don't wait to sign defensive pacts*"
cdtw.6	"*if we're going for a space victory let's quietly relocate our capital away from the coast*"
	Probably bugged in LoR SDK ModMod because `AI_isDoStrategy` had been confused with `AI_isDoVictoryStrategy`.
See also	[kekm.19](#) disallows moving the capital once the spaceship has been launched. So the AI can only move its capital prior to launch.
cdtw.7	"*if going for culture victory, build lots of fighters to avoid be*[ing] *blitzed*"
	Disabled again. I guess that's really only smart when playing with Dale's Combat Mod.
cdtw.8	"*if our best city attacker is crappy, be less likely to build invaders, unless we aren't paying support costs yet*"
	Disabled again; K-Mod seems to have this covered.
cdtw.9	In Unit AI, a few checks for same owner replaced by same team.

advc.rom	Misc. changes adopted from RoM-based mods
Credits	Some of the largest Civ 4 mods are based on [Rise of Mankind](#) (RoM, started 2007), including [A New Dawn](#) (Rand), [Caveman to Cosmos](#) (C2C), [Vincentz Infinite Projects](#) (VIP) and [Dawn of the Overlords](#) (DotO).
	RoM/Rand and C2C have had several AI programmers among their contributors. It seems that most of the AI changes are specific to the many features added by those mods, superseded by K-Mod/AdvCiv or not worth adopting from a cost-benefit angle.
	I'm listing some of the changes that I did adopt here; others are so minor that I'll only tag them with "advc.rom" in the source code.
See also	[107](#) and [110](#) use a bit of C2C and RoM code. [advc.ctr](#) (AI) uses a bit of RoM code.
Tbd.	I'm still in the process of sifting through the revision histories on SourceForge. The larger part (something like 1000 revisions) is still to be done.

rom1	*"% modifiers are undervalued by the AI for two reasons:* *1) It only calculates its effect at the time of construction, but that discounts future increases as base research goes up* *2) In the early game because this calculation only takes account of the absolute increase, not its proportion of the civilization whole, which will be substantial early on (indeed 100% with 1 city).* *We therefore apply a small boost to reflect futures, and compare to the whole and make an upward adjustment proportional to the ratio with the civilization whole.* *FUTURE - should we do this for other commerce types too? I am inclined to say no for hammers (since what you produce with them scales with civ size, which the tech tree does not except very indirectly via progression through it). Not sure about gold or espionage."* SVN revision This change increases the utility assigned by the AI to buildings that increase a city's research rate. However, this applies only when the City AI wants to focus on research buildings, which is actually never the case in K-Mod. So I'm also adding this koshling change from the same revision: *"increased priority for economic builds,"* which makes the AI choose city production with focus on all economy buildings, including research buildings. Removed this again; in multiple test games, it never had any effect. Looks like K-Mod already prioritizes economic buildings sufficiently. The AdvCiv code with the change still present: Git commit
Credits	Koshling
rom2	AI doesn't reassign working plots while in anarchy as "*plots return no yields anyway*." SVN revision Reassigned working plots don't seem to be causing problems in AdvCiv, but it does save some CPU time.
Credits	Afforess
rom3	Resources can be randomly discovered only on terrain where they can normally be placed. E.g. no Gold on Grassland.
Rationale	The particular terrain restrictions for Gold, Silver and Gems are pretty nonsensical, but slightly helpful for game balance, and should be enforced for consistency.
Credits	Afforess (but merged from this SVN revision by alberts2)
See also	129 changes the terrain requirements for Gold and Silver a bit (but they still don't make sense).
rom4	Avoid a costly canTrain calls in CvCityAI.
Credits	alberts2 for Caveman2Cosmos; SVN revision
See also	In part superseded by the `FOR_EACH_NON_DEFAULT_PAIR` macro (advc.enum)

advc.rstr	Minor improvements for ranged strikes. For mod-mods.
Credits	Inspired by keldath's DotO mod. See the end of this post and subsequent posts about ranged strikes in DotO. I haven't adopted any major changes though; just small improvements (in a strict sense).

advc.mnai	Various tweaks to the behavior of AI units from the "More Naval AI" mod

Credits	Too laborious to document those separately. Fully or mostly merged: Git commits 1 2 3 4 5 6 Partly merged: 1 2 3 4 All by tholal. I've scoured the whole repository. There are some more things worth adopting; the links to those commits are scattered across the manual. More recent changes from MNAI-U ("unofficial" continuation of MANI) by lfgr: 1 2 (more to come probably, eventually)
See also	Some other changes from MNAI have different tags; will have to do a full-text search on the manual to find them.

devolution	Contributions by Erik (cf. this post and subsequent ones in the AdvCiv thread)
See also	advc.make: Build optimizations 003h: A performance tweak of his that I merged from the "We The People" mod
Crash while exiting to desktop fixed (link).	
OPT	Performance tweaks; merged: Git commit
BUG	Bugfixes; merged: Git commit Also merged this correction by vedg (Igor). And I've fixed a similar problem in `CvPlayerAI::AI_techValue`.
AI	Misc. AI tweaks; merged Git commit 1 2
BM	Benchmarking functionality added to AI Auto Play: Git commit I've added a new shortcut Ctrl+Shift+B to avoid confusing players who just want to watch the AI play. Note that this measures wall clock time, not just the CPU time spent on the `Civ4BeyondSword.exe` process. Measuring the latter doesn't really seem possible in Python 2.4 (and generally difficult on Windows). That means, applications running in parallel can significantly skew the result, especially video playback in my experience.
See also	004y adds the shortcut to the list in Civilopedia.

Work in progress

First off, misc. open issues that, ideally, I would have liked to address before v1.0 (ordered by priority). While a lot of issues remain, I did take care of (and deleted) all issues that were really bothering me and were actionable. Above, the appendix lists many more open issues ("Tbd." boxes) of low priority. (I've also been maintaining a long list of small quality-of-life changes of even lower priority that I'm not even going to include in this manual.)

- [UWAI] Perhaps add some war utility when an AI civ hasn't seen a war started in a long while (or never) and, conversely, subtract some war utility when many wars have been declared lately. Not a rational consideration, just for the right amount of drama and opportunity. Could perhaps check AI diplo memory (`MEMORY_DECLARED_WAR`) instead of adding new AI variables. CFC post (2nd quote box)

- [031] Can I get the AI to place fewer cities that really bother players who think that long-term city planning is more important than high-level players do? That seems to be at the bottom of continued complaints about AI city placement, especially about cities one tile away from the coast. (Though I could be wrong and there might be some unintended AI behavior at work – however, the AI cities I see in my games look reasonable, at least when viewed from the perspective of their owner at the time that they are founded.) Should be possible to avoid cities that will irk players without changing the overall approach. (Favoring quality of worked tiles over quantity is a fundamental insight of advanced-level players; this is not a case of the AI being programmed to be too fancy. Should probably add a loading screen hint to that effect, e.g. advising players to count the good tiles rather than the bad tiles – which should give them a clue that the AI behavior is intentional.)

  Will have to run a bunch of AI Auto Play games, preferably on Huge maps, to find inland cities that are able to work any coastal water tiles with extra yields (seafood, Colossus, Financial trait). Then see if these are primarily caused by insufficient exploration or by a low priority for sea access. In the former case, one consideration is to discourage cities with – or even just adjacent to – unrevealed workable tiles. Though getting the AI to explore more effectively would be preferable.

  CFC post (I've also received a private message lately in the same vein.)

- [124] An AI leader should refuse to sign (or should cancel if already signed) Open Borders with a leader who is the worst enemy of several other leaders that the first leader cares about. May have to weigh the number of foreign trade routes that the first leader is foregoing and the importance of (appeasing) the second leader (city count?) with the first leader's attitude toward the third parties and their importance. Reason for trade denial: Could use "We couldn't betray our close friends!" (`DENIAL_ATTITUDE_THEM`), but probably better to add a new denial type, e.g. "You have made powerful enemies..."

- [advc.inl, advc.enum] Some large classes – CvPlayerAI and CvGame come to mind first – could use a pass of refactoring that replaces arrays with enum maps, `for` loops with various macros and iterator and allows all small functions to be inlined (and not just when whole-program optimization is used). Note that CvInitCore, which some CvGame functions forward to, is in the precompiled header, so CvInitCore functions can be called from `CvGame.h`.

  003t: XML loading code should also use enum maps, especially `CvInfo_Unit.h`.

- [UWAI] Affection cost in team games should be based on the no-war attitude probability of the current team member, not the average of all team members. [Related discussion](#)

  Problem with this: Call locations of `CvTeamAI::AI_noWarAttitudeProb` should be consistent with the behavior of UWAI's `Affection::evaluate`.

- [UWAI] Dial up the impact of war success on the `conf`... variables in `InvasionGraph::Node::step` when a war has been going on for some time and has seen a lot of fighting (high total war success). Currently, a human player attacked by an AI civ with clearly superior power ratio (say, 150 to 200%) and superior production capacity won't ever get a peace deal. In such a situation, the human civ may well be able to hold onto its cities, but probably won't manage to start a counteroffensive. Even if the AI will win such a war in the long run, it won't hurt to at least take a 10-turn break now and then.

- [UWAI] The AI estimate of military build-up based on the power graph needs to account for units destroyed during war somehow. Those losses do not imply that a civ has stopped producing new units.

- [sha] Rank-based AI relations penalties shouldn't abruptly go from -2 to 0 when an AI leader (e.g. Montezuma) falls very far behind. The score difference needs to result in a multiplier.

- [130i, 130p, 148] Some of the Dynamic Diplomacy changes have made it more difficult to befriend AI players. Consider letting the relations modifiers from Open Borders and Defensive Pacts increase to up to +3. As for Defensive Pacts, the AI may then also have to be more discerning about signing them or the relations modifier will have to be based on how threatened the AI feels and who is protecting whom. Fair trade could be based more on the total trade value; it's still mostly based on the difference in trade values. In AdvCiv 1.0, I've already decreased the Friendly threshold by one more and made the mutual struggle diplo modifier some 20% more sensitive to war successes (and thus also a bit more lasting). I still don't think this fully solves the problem. In any case, having more ways of pleasing the AI would make games without tech trading more interesting. CFC posts: 1 2

  [130i] Pleased attitude might be too difficult to reach in team games, perhaps because of my changes to the Open Borders relations modifier. [CFC post](#)

- [130r] Exponential backoff mechanism for AI contact delay? To ensure that the AI doesn't offer the same trade over and over. [CFC post](#)

- [UWAI] War trades might be offered too rarely. Perhaps I should relax some attitude thresholds ([CFC post](#)), or perhaps it's already better now (since v1.0) that Friendly attitude is easier to reach. v1.02 also tweaks the war utility threshold for the "enough on our hands response" (but I don't think that'll make much of a difference).

- [912] Add a colony maintenance modifier in XML. Maybe use it to let Mercantilism reduce or disable colony maintenance. It would be a bit anti-synergistic though because maritime empires benefit especially from the Free Market extra trade route. [CFC discussion](#)

- [BM] Consider limiting the `Civ4BeyondSword.exe` process to a single core for performance tests; maybe even in general if it turns out to be faster. Can try it through the Task Manager: "Details", context menu, "set affinity". Programmatically: see 2nd answer [here](#).

- [055] Global Warming: Should the population portion of the formula be tied to technological progress (or is it already), e.g. at least one player in the Industrial era? Some problem with map size or speed scaling? CFC discussion play report (with some sample numbers)

  Future Tech should grant some Global Warming protection. Idea (CFC)

  Perhaps add a "No Global Warming" game option.

- [200] Is the (K-Mod) Great Artwork ability too powerful when starting in a later era (since no culture has accumulated yet)?

- [001] Check if the Ctrl+Shift+L shortcut (Civ Changer) disables the "wait at end of turn" player option as reported here.

- [kekm.3] When a civ declares war and both sides have Defensive Pacts with (different) third parties, then the third party that has a Pact with the aggressor should not end up at war. Verify that this is what happens.

- [021b] PerfectMongoose: See if I can combine the PW2 and PW3 elevation maps for the best of both worlds. Mountain ranges look strictly better with PW3, whereas PW2 produces larger and more convex continents, which tend to play better than the Perlin noise curls.

- [077] Perhaps show info about foreign trade routes ("imports") on the Info tab of the Foreign Advisor screen. CFC discussion

- [312] Consider reducing the XP modifier on the attack from 4 to 3. This is already implemented for Barbarian units. CFC post

- [010] Perhaps re-enable worker stealing and add a tech requirement. CFC discussion

  The best fit at the right time appears to be Monarchy; not very flavorful. Perhaps better to move Slavery to Iron Working (CFC discussion) and let Slavery enable worker stealing. If an icon is needed for the tech tree, use the Worker button graphic (or better: the active player's unit of the Worker class, default unit if no player active). Or simply create a "No Worker Stealing" game option under "No Slavery"?

  Accompanying AI change: When losing badly at war, too many workers overall and too many assigned to a local city, then disband 1 local worker (only in the city tile?) per turn. May disband another if evacuating. Should probably keep 1 worker assigned per city in any case. See also *Tbd.* under 010.

- [031] The AI might value strategic resources too highly when it has few cities and no prospects for expanding much; and perhaps not highly enough in the late game. The evaluation doesn't really take into account how many cities will benefit from a resource. See comment in `AIFoundValue::nonYieldBonusValue`.

- [082] The AI shouldn't embark units while gathering them for a naval assault. CFC discussion
  Might it be enough to remove the `AI_load` call in `CvUnitAI::AI_attackCityMove` under "*if no land path to enemy cities, try getting there another way*" when `pTargetCity` is `NULL`? Or perhaps the loading happens somewhere else entirely most of the time …

  Potentially helpful: Git commit from MNAI

- [004] Some of the game text additions proposed in this post by CFC user crullerdonut for terrain improvement should be worth implementing. I'd probably want to be a bit

more low-key, and some of it shouldn't appear only in Civilopedia. [This](#) post also flags the lack of information about Tundra Farms.

- [UWAI] If add another game option, consider replacing "Aggressive AI (Legacy)". (Should then also change the prefix of loading screen hints from "UWAI" to "AdvCiv".)

- [UWAI] Too little warfare in team games? That may not be so wrong as AI teammates aren't good at coordinating their attacks; so even when two members of one team are on a landmass with just one member of another team, starting a war isn't a no-brainer. Risk aversion might be a problem – if one teammate expects to gain 2 cities and the other to lose 1, the team typically won't go to war. Could simply increase utility from GreedForAssets artificially to get more wars going. That said, lately, there didn't seem to be that little war fare.

- [advc.ctr] One player has reported that the AI will occasionally give away cities too cheaply; not sure if there's really a problem.

  Make some leaders extra thankful about city liberation, e.g. Gandhi. (Maybe also some slightly less thankful.)

- [002] On certain surfaces, the Farm graphic flickers during camera movement. I haven't checked yet whether this also occurs in BtS. The part that flickers is a flat rectangle that seems to be related to the shadow cast by the barn. Perhaps it's too close to the ground; or maybe I can just throw that out somehow? Even that is difficult to do without any knowledge about `.nif` files ...

- [101] A couple of users report that it takes too many units or too much time to supress revolts, especially prior to Rifling. Consider buffing Musketman. 1 (in the middle) | 2

  Perhaps foreign culture strength should be reduced a bit when the cultural owner has been defeated.

  Consider slightly adjusting foreign culture strength or revolt probability to the map size. If there is more space to fill, there should be fewer incentives against rapid expansion. CFC discussion

  (v1.0 has addressed some of these issues, but maybe not fully – has made revolts harder to suppress in the late game, but not easier in the early game; foreign culture strength gets adjusted to the difficulty level of the city owner.)

- [300] Disable the trade abilities of Sailing and Astronomy for Barbarians. Cf. *Tbd*. under id 300. CFC post (with savegame; can reproduce the situation by letting the Barbarians reconquer the city of Kazakh).

- [055] Make it easier to disconnect Coal and Oil on purpose (for the sake of avoiding health penalties and Global Warming anger/ events). Current problems: Can't discon-nect resoues in city tiles; disconnecting Coal and Oil makes it impossible to build Railroads; automated workers may reconnect disconnected resources. It seems that trading a resource away works around all of these issues (not sure about Railroads), but there isn't always a taker. Also, both trading and self-pillaging are pretty counterintuitive plays. Might be best to add buttons somewhere for toggling Coal and Oil on and off, e.g. on the Environment tab (Economics Advisor). Disabling a resource that way should arguably also cancel any deals importing the respective resource or, if impossible to cancel, notify the player of the ongoing deal. Another idea is to turn the resource icons on the city screen into buttons that allow any

resource to be toggled on a per-city basis. Ctrl+click could affect all cities. Hover text should arguably only mention those actions for Coal and Oil. Disabling a resource could perhaps also affect tile yields, treating them as if the resource didn't exist. (But that wouldn't work cleanly on a per-city basis.) Related [CFC post](#) (2nd spoiler box)

- [[139](#), [UWAI](#)] Can `CvCityAI::AI_updateSafety` take into account remaining movement points and turn order? Currently, when negotiating peace, the AI is equally alarmed about units that can immediately take the city and units that will be able to do so on their next turn. Both is alarming, but not quite the same thing.

  The AI should not just care about cities that are all but certain to fall. Need another `CitySafetyTypes` enumerator: `CITYSAFETY_TOSSUP`.

  Related [CFC post](#)

- [[706](#)] When the active (human) player gets defeated in a Rise&Fall game, another player (typically player 0 from the initial chapter) becomes active when the defeat and civ selection popups are shown, and this can give away part of the map. Not sure how to prevent that. [CFC post](#)

- [[116](#)] Should use the `AI_assetVal` function (introduced for city trades) also for raze decisions.

- [[703](#)] When loading a game or returning to the main menu, `CvVictoryScreen.iScreen` in Python should be reset. Perhaps this could be piggybacked on some existing dirty-bit. As it is, the R&F tab may remain active after loading/ starting a non-R&F game.

- [[036](#)] Perhaps modify the trade value that the AI assigns to gold based on the attitude toward the civ that pays the gold. [Rationale](#) (CFC); [another](#) related post

- [[036](#)] To evaluate techs that enable a new terrain improvement, at least during the Ancient era, when performance isn't an issue, the AI should go through each city, check for tiles in the city radius where the new improvement can be built and compute the difference in yields between the new improvement and the worst currently worked tile. And then put the sum of those differences in relation to the total yield rate of all its cities.

  [CFC discussion](#)

- [[130w](#), [148](#)] The relations penalty for having capitulated vassals should decrease to -1 per vassal eventually. Based on the time since capitulation I guess. To be consistent with the "oppose your ruthless expansionism" penalty, which tends to decrease over time. May then (perhaps) want to revert an increase of the expansionism penalty implemented in v1.01.

- [[UWAI](#)] When evaluating a sponsored war, a phase of military buildup should be assumed for all war parties because the hireling usually won't be ready to invade directly. The assumed duration could be something like 10 minus 1.5 times the tech era of the hireling. Also, the `attackerUnprepared` flag that already exists in `InvasionGraph::Node::step` should take into account whether it's a sponsored war.

- [[UWAI](#)] The limit for what the AI is willing to pay for peace should be increased when the tactical situation (clearly) favors the other side. Normally, it doesn't make sense to pay much for 10 turns of peace, especially not to a human player, but it does make sense to pay extra if it gets the AI out of a bad spot.

- [[UWAI](#)] `InvasionGraph::Node::step` sets its `isNaval` flag solely based on whether the

target city is reachable (at all) by land. Worse, `UWAICache::City` doesn't store separate distances via land and via sea. `UWAICache::City::updateDistance` mustn't just set `pwd=d` in the `canTrainAnyCargo` branch. Should instead maintain a separate vector `pairwDurationsLandOnly` for distances computed assuming that `canTrainAnyCargo=false`, and store an additional distance value `distByLand` at the `UWAICache::City` object. Try encoding non-reachability through a distance value of -1.

Then `Node::step` could set `isNaval` if either there is no land-only path to the target city (as is the case now) or if (important) `WarEvalParameters::isNaval()` and the distance by sea (i.e. by any means; sea or land) is significantly shorter than the land-only distance. Also `UWAICache::City::attackPriority` will have to use the land-only distance for cities reachable by land if `WarEvalParameters::isNaval() == false`. Will probably need a separate comparator `UWAICache::City::byLandAttackPriority` for this.

- [UWAI] `InvasionGraph::Node::resolveLossesRec` seems to use costly memory allocation.

- [UWAI] Limited war might be used too often and total war too rarely based on one user report. My own observations are inconclusive; the AI behavior is probably not way off.

- [130v] Now that vassals generally can't sign peace treaties and don't get a peace treaty from city trades (advc.ctr) either, I could allow voluntary vassals to hire war allies (as in BtS). Originally, my concern was that the vassal would force its master into a peace treaty through change 146.

- [130m, 130w, advc.sha] Through human attacks, the AI relations bonus for "mutual struggle" (`CvPlayerAI::AI_getShareWarAttitude`) can change throughout a human turn. This can lead to an outdated AI attitude cache. Try updating the mutual-struggle component of the cached value via `CvPlayerAI::AI_changeCachedAttitude` before/after every human attack (for all war allies of the human). Also, for war enemies, `AI_getWarAttitude` and, for all rivals, `AI_getExpansionstAttitude`, `AI_getRival VassalAttitude`. (The latter two both call `AI_expansionistHate`.)

- [advc.sha] Perhaps bring back the relations bonus among civs in the lower half of the scoreboard for AI leaders with rather peaceful personalities. (Based on peace-weight? But what about Augustus?) Not during the game's start era though (ranks change too often).

  CFC discussion

- [130p] Perhaps becoming an AI civ's worst enemy by trading with the current worst enemy shouldn't be entirely ruled out. It's not necessarily illogical. This would concern the `iAttitudeDiff` formula in `CvTeamAI::AI_enemyTradeResentmentFactor`.

- [700] When playing with the R&F option, the "Retire" button in the Retire popup should be labeled "End the chapter" instead.

- [UWAI] Try implementing two simulation trajectories (one slightly optimistic, one slightly pessimistic) when performance isn't an issue, e.g. so long as there are fewer than 100 cities in UWAICache. In the late game, outcomes tend to be clearer (greater disparities in military power) anyway.

  The constructors of MilitaryAnalyst and InvasionGraph should be the place to implement this.

[CFC discussion](#)

- [[130f](#)] Peace treaties should block embargoes. I think, currently, players could ask the AI to stop paying reparations to a third party.

- [[UWAI](#)] Should perhaps assume a higher guard ratio (in `InvasionGraph::Node::step` and in `ArmamentForecast.cpp` – I've already tweaked the latter, but could tweak it more by removing the current special treatment for humans entirely) for humans because, currently, wars against humans are usually judged by the AI as all (take several human cities) or nothing, which takes away the nuances from AI war evaluation. Possible side-effects: AI could become less alert about human attacks, less willing to pay for peace, less willing to attack in the early game (when a high guard ratio matters most). That might be OK.

- [[UWAI](#)] To (further) encourage AI civs with a very powerful military to fight multiple wars at once: If all war enemies (current and projected) are pushovers (`UWAIAgent::Team::isPushover`), force `weId` in `InvasionGraph::Node::findTarget` to target `WarEvalParameters::targetId()` and force `target=NO_PLAYER` if `InvasionGraph::isPeaceScenario`. To evaluate and implement this, I'll need a game state where a civ that is way ahead militarily is being too slow mopping up.

- [[104m](#)] When asking for help, the AI should check if a package of several human surplus resources would be valuable enough. (Also: Changes to tribute and help requests don't really belong under id 104m.)

- [[700](#)] When the R&F option is enabled, perhaps show the victory stage in the scoreboard help. Only show it if it's greater than 2. Maybe as a roman numeral after the score, e.g. "1744(III)", colored orange if it's a rival at stage III, red if a rival at stage IV and green if it's the active player at stage III or IV. And, ideally, the stages should be computed by a simpler non-AI function that uses mostly public information.

- [[083](#)] Should add a shortcut somewhere in `CvUnitAI.cpp` that makes the AI conquer empty cities in reach, perhaps just with a single unit – hoping that the rest of the stack will respond appropriately to the conquest. The AI currently insists on going to an attack-from tile chosen in advance (not necessarily on the shortest path) before entering the city. Possibly related CFC posts [1](#) [2](#)

    Similar problem: AI transports won't make opportunistic attacks on lightly defended cities while headed toward their target city. The selection of the target city may also be too predictable. [CFC disucssion](#)

- [[UWAI](#)] Try letting an AI civ estimate human military power more highly in situations when the AI civ is not itself at war with the human. This should avoid AI-on-human dogpile wars that are supposed to keep human cities from falling into a rival's hands. It should also make the AI a bit braver when fighting alongside a human (more ready to try and snatch away territory from a shared enemy) and more alarmed about human wars of conquest.

- [[306](#)] Barbarian ships on patrol should eventually just permanently blockade some city, or at least for a long time. Through `CvUnitAI::AI_patrolMove`, `CvUnit::getGameTurnCreated` and a coin flip.

- [[UWAI](#)] `WarUtilityAspect::lossesFromBlockade` should be generalized to also cover air bombing.

- [[UWAI](#)] `MilitaryBranch::Logistics` should store military power and cargo capacity

separately. Currently, `Logistics::power` returns the cargo capacity.

I guess this'll mean adding a `capacity` function to MilitaryBranch and all its subclasses. Then go through all occurrences of `[LOGISTICS]` in the code and replace `power` calls with `capacity` where appropriate. Logistics power should continue to be included in Fleet power.

- [550] The relations bonus from "shared discoveries" should be based on the total cost of the traded techs.

- [550d] The research cost adjustment for team games (`bTeamSizeModifiers`) in `CvTeam::getResearchCost` doesn't seem strong enough. May have to apply a modifier to all research costs (not based on individual team size) in addition – if team size is punished even more, Permanent Alliances may become unattractive. Could e.g. multiply research costs by `sqrt(CvGame::countCivPlayersEverAlive() / countCivTeamsEverAlive())`. Apply this factor regardless of whether the `bTeamSizeModifiers` parameter is set (I guess).

- [082] Implement `UWAICache::updateRelativeNavyPower` (see comments there) and an auxiliary function for the intel ratio. Currently, the AI magically knows how a civ's military power is split between land and sea units. The BBAI function `CvTeamAI::AI_getRivalAirPower` also has this fault. Change 650 also lets the AI cheat with the number of enemy nuke units.

- May want to force-disable `HAPDebugger` through `gDLL->ChangeINIKeyValue` in final-release builds. Causes the game to get stuck at launch unless the HAP debugger is actually running. Only a handful of modders have ever used it, and players sometimes enable it out of curiosity.

- [085] Icons (e.g. religions) on the scoreboard are a bit displaced toward the top. I took a quick look at `CvMainInterface.py` and `BUG Scoreboard.py`, but there is no obvious problem. Perhaps an issue with the GameFont file.

- [707] Not sure if R&F games will end properly on a time victory. Should be tested once with retiring (causing the game to end during Auto Play) and once without (player controls a civ when the time limit is reached).

- Perhaps add an optional unit action button that deselects half of the currently selected group. This has been requested here (penultimate bullet), and is actually pretty easy to implement, as pointed out in this post.

- [127c] (see there under *Tbd*.) It's sometimes (very rarely?) impossible to add gold to the trade table. I've seen this bug reported for Realism Invictus and Dawn of Civilization as well. Workaround is to restart Civ; reloading without restart might also suffice.

- [UWAI] Do something with `AIDeclareWarProb` from `Civ4HandicapInfos.xml`; currently, it's unused. Just delaying AI war planning isn't good - that may well lead to bigger attacks when they do happen. Perhaps make it into a preference for limited over total war.

- [133] If resource trades are still canceled too frequently, try increasing the inertia for keeping resource trades alive further. That said, it's probably already only canceling when "nothing to gain", so a tolerance parameter may have to be added to the denial function `CvPlayerAI::AI_bonusTrade`. Or simply reduce the cancellation probability even if "nothing to gain" (when trading with a human). For AI-AI trades, consider

applying the inertia code that currently applies only to Human-AI trades – canceled AI-AI trades could have a domino effect (unclear).

CFC discussion 1 2

- [102] To be tested: Enable "Show friendly moves" in networked multiplayer with simultaneous turns while the other human player is in Globe view. The result should be that the first player sees friendly moves (despite the other player's Globe view, which normally skips foreign moves) and that the other player does not see friendly moves -- but I guess there'll have to a delay while the moves are shown to the first player. Is this working correctly? What if both are in Globe view? The clauses I've added under the comment "Are these timers synchronized" in `CvSelectionGroup.cpp` might cause a delay in that case (although no moves are shown to either player). What if I remove those clauses? How about non-simultaneous turns?

- Had a crash once right after generating a map using the (abandoned) TrueStarts map script with default settings (initial map; not regenerated). One of the `operator=` string functions was called from the EXE; apparently one of the strings was a bad pointer/ reference. If this happens again, I need to check the contents of any valid strings in the debugger's context. I also remember one crash after/ upon map generation with the AdvCiv-based Dawn of the Overlords mod (no debugger attached). I've been starting tons of games, so this seems to be a very rare bug.

- [123f] To be tested: Is fail gold for world projects (the Internet) working correctly?

- [kekm.25] To be tested: Do vassals and friends behave correctly in repeal votes? Are human vassals correctly forced to vote?

Larger features that I had considered including before v1.0 (not ordered by priority):

- Merge further UI conveniences from BULL-based mods and PlatyUI: See "*Tbd.*" under change id 004.

- Adopt some misc. tweaks from the BASE mod. Or rather from the mod's changelog (in German) – the DLL is closed-source.

- Adopt the improved savegame format (explanation | Git commit) from the "We the People" mod and perhaps also the revised setup for translations. Or at least this little change. And UTF8 support. (Or perhaps that's easier to adopt from "A New Dawn 2" – `UTF8Support.h`/`cpp`). Currently, special characters in leader and city names aren't (always) displayed correctly if they're not HTML-encoded (Windows 1252 encoding, to be exact) in the game text file. And the Japanese (Russian?) edition of Civ might not even be able to load a mod with certain special characters next to an XML angle bracket.

  When changing the savegame format, it would also be a good time for transitioning to a single (31-civ) DLL.

- BBAI includes a file `unofficial patch list.txt` with "outstanding issues". Most of it is incomprehensible, but a couple of items seem worth looking into.

- Scour SVN revisions from the the early days of Caveman2Cosmos and revisions from RoM-AND from 2010/2011 for AI improvements and other useful tweaks that are easy to adopt.

- Complete the EuroWorld scenario. Still a lot to do though and I've run out of steam

because the areas that interest me most are already done and because I regret a few of my choices regarding landmass proportions.

- In large games, human players get contacted too frequently by the AI. Ideally, before contacting a human player with an offer, the AI should compute some measure of confidence about the offer being of interest to the human and decide based on that and the number of players known to the human whether to proceed. Since offers can't hurt the human player, there could be an option on the BUG menu in addition to further discourage AI-to-human offers. I don't think the requests that come with a diplo modifier are a big nuisance, so I don't think that a "cease bothering us" option is ultimately needed. Known unhelpful offers: Trade offers for dead-end techs like Divine Right; repeated offers for Open Borders. CFC disucssion

- Move AI-to-human diplomacy to the start of the contacted player's turn. See "*Tbd.*" under 001e. CFC discussion

- To further improve the performance of UWAI in games with far more than 18 civs, implement a heuristic for pre-selecting only the top 12 (or so) war targets. Top priority: current war enemies, then humans. Then, for a start, I could try using K-Mod's `CvTeamAI::AI_startWarVal`. For the targets that are ruled out this way, any calls to UWAI functions need to be avoided; in particular, `WarEvaluator::evaluate` and `UWAICache::update` mustn't be called. I might get away with updating the set of relevant targets only once per turn. When a decision on war and peace comes up suddenly, e.g. a war trade, war/peace vote or a tribute demand, K-Mod code could be used as a fallback, but it's probably better to use some safe default behavior instead, e.g. refuse categorically to be hired for war. Another potential issue: Evaluating a war plan against a civ that has other war enemies could be problematic if no UWAI data is available about those third parties.

- Try removing the at-war checks in the code for AI-AI war trades (`CvPlayerAI:: AI_doDiplo`), i.e. let the AI sponsor wars without declaring war itself. UWAI should be "smart" enough to handle that. Performance will be a problem though; may have to check a bunch of preconditions beforehand to save time, e.g. check `UWAICache:: warUtilityIgnoringDistraction`. And can always make the check probabilistic to improve the performance by a constant factor.

   Brokered peace might also be doable. And the AI could demand that a human player make peace.

- AI civs should ask other AI civs to stop trading with disliked third parties. Precondition: Must be at war with the third party or the third party must be the worst enemy. Then check denial (`CvPlayerAI::AI_stopTradingTrade`). Need to add a function `AI_stopTradingThemTradeValue` that says how much the AI wants another civ to stop trading. Embargoes will have to be win-win in terms of that value and `AI_stopTrading TradeVal`. Could base the contact frequency on the `ContactRand` value for `CONTACT_STOP_TRADING` (currently used for AI-to-human stop trading requests).

- Extend advc.xmldefault to some of the the large XML files like `Civ4UnitInfos.xml`. This would make XML changes post-v1.0 more convenient.

- Improve the AI code that decides where to gather units for war or where to park them when no war is planned (`CvUnitAI::AI_moveToStagingCity`). Avoid putting them all into a border city with poor defenses.

- Make third parties less supportive of wars against their partners:

- – Disallow attacks inside the borders of a non-war party. Air attacks should remain allowed. Not sure about attacks at sea.

- – Perhaps restrict railroad movement in third-party borders somehow during a war (or always unless sharing a war, defensive pact or vassal agreement).

- – Being at war with a friend (Friendly attitude) should raise the AI attitude threshold for Open Borders by one level. There may have to be some delay to avoid making an AI civ that starts a war look bad when its invasion is immediately stopped by canceled Open Borders. The friendly AI civ will also have to check if the aggressor is not too dangerous (e.g. check embargo denial).

- Tweak the AI personalities to make them more distinct and to match the historical background better. (But don't introduce new quirks that players have to be aware of.) I have notes for about one third of the leaders. May also have to adjust some diplo texts to fit the personalities.

- Module with graphics from Varietas Delectat (VD) or BAT. Not sure what the best source would be. Leoreth has made some adjustments to the VD graphics that would be nice to have. Anyway, I won't do this on my own as I don't even want to play with those graphics. If someone else does a merge, I can look into making the process easier. CFC discussion; see also *Tbd*. under 002j about some changes to unit graphics that I'd like to make.

  A different approach would be to curate graphics (not just from VD, graphics modders have kept busy to this day) that won't have to be optional because they're not just more realistic but also more functional than the original graphics. But, even then, the increased download size will be a downside. And it's difficult to be more functional than graphics that players are highly familiar with.

- Add a (worst) "enemy of my enemy" relations bonus.

- Perhaps decrease the 300% modifiers on Marathon speed to 250%.

- Changes in preparation of a tech diffusion system (see also "city trades" above). Should only apply when tech trading isn't possible.

  - – The AI should make more gold available for trade.

  - – The AI should accept gold-per-turn as payment joining a war (now that joining a war results in a peace treaty due to change 146).

  - – Gifted units should result in a sensible relations bonus (based on whether the receiving side needs the units) and penalty from war enemies of the recipient.

  - – Increase the trade value of gold to make payment in gold (e.g. for civics changes, joint war) more affordable.

  - – Make the "fair and forthright" and "traded with our worst enemy" modifiers more sensitive. And also the "years you've supplied us with resources" bonus.

The documents about future gameplay changes on GitHub are mostly intended for a version 2.0 of this mod (and, as such, unlikely to ever be implemented). The following changes would've been suitable for v1.0 (or, say, 1.1):

- Various changes concerning religions, the Apostolic Palace and UN

- Forced Labor civic to replace Slavery when playing with the "No Slavery" option (through an XML tag list of game options required to be enabled or disabled in

`Civv4CivicInfos.xml`).

- Nerf some food resources
- Add an Islet terrain feature
- Don't require Flanking I for the Navigation I promotion
- Optional tech diffusion from tile culture
- Optional transaction cost for tech trades
- Limited Worker charges
- Drafting may consume stored food
- Changes to city conquest rules
- Reduce process conversion rates (e.g. Wealth), perhaps based on difficulty level.
- Penalize extreme commerce slider positions (or frequent changes; cf. 157)