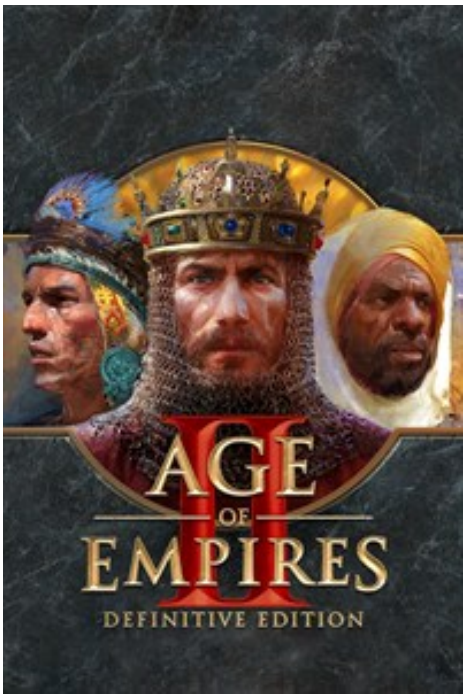
Age of Empires DE, a tutorial for modding

Koldar

Version 1.0.0



Date: December 31, 2020

Contents

List of Figures **iv**

List of Tables **v**

Acronyms **vi**

About this report **vii**

1 Introduction **1**

2 Background and Related Works **2**
2.1 Maps 2
2.1.1 Types of maps 3

3 Basics and Definitions **5**

4 Creating a basic mod (MWE) **8**
4.1 Publish mod 10

5 Implementing other mods using Advance Genie Editor **12**
5.1 Altering corpses decaying 12

6 Creating Random Maps using Random Map Scripting (RMS) **13**
6.0.1 Filling PLAYER_SETUP 13

Appendices **14**

Semantics of `Age of Empires II: DE Exe` **15**
.1 BattleServer 15
.2 certificates 15
.3 Docs 15
.4 Schema 15
.5 Support 15
.6 Tools_Builds 15
.7 webclient 15
.8 wwise 16

.9 resources . . . 16
.9.1 _common . . . 16
.10 widgetui . . . 16
.10.1 atlas . . . 17
.10.2 backgrounds . . . 17
.10.3 campaign . . . 18
.10.4 ingame . . . 18
.10.5 menu . . . 18

Advance Genie Editor **19**
.11 Required Knowledge . . . 20
.12 Units parameters . . . 21
.12.1 Language Files . . . 23
.12.2 Graphics . . . 23
.12.3 Statistics . . . 23
.12.4 Projectiles . . . 23
.12.5 Attributes . . . 23
.12.6 Sounds . . . 24
.12.7 Tasks . . . 24
.13 Graphics . . . 24
.13.1 Deltas . . . 25
.13.2 Deltas . . . 25
.13.3 Angle Sounds . . . 25

RMS file format and layout **26**
.14 RMS control flow . . . 27
.14.1 Conditional flow . . . 27
.14.2 #include_drs . . . 27
.14.3 #const . . . 27
.15 RMS file sections . . . 28
.15.1 PLAYER_SETUP . . . 28
.15.2 LAND_GENERATION . . . 29
.15.3 CLIFF_GENERATION . . . 34
.15.4 TERRAIN_GENERATION . . . 35
.15.5 CONNECTION_GENERATION . . . 38
.15.6 OBJECTS_GENERATION . . . 41
.16 Functions . . . 45
.16.1 Random . . . 46
.16.2 Choose a randomly chosen scenario . . . 46
.17 Inherited constants from the environment . . . 46
.18 Popular included files . . . 47
.18.1 thebr_setup.inc . . . 47
.18.2 F_seasons.inc . . . 47
.19 Constants . . . 49
.19.1 Terrain Type . . . 49
.19.2 Map Type . . . 58

Terms **60**

Bibliography **61**

List of Figures

2.1 Example of a cliff in a map. 3
2.2 Examples of maps available in Age of Empires II: DE. Red, Yellow, Green and Blue diamond represents players positions. If a team game is played, Red and blue are allied against team yellow and green. A shoe represents a nomadic map. 4

4.1 Image to put as background 8
4.2 My mods window 10

2 Background menu examples 17

3 Standard Advance Genie Editor window 20
4 Standard Advance Genie Editor window. The red box highlights the category tabs grouping the parameters you can tweak according to the Age of Empires II: DE field. 21
5 Standard Advance Genie Editor window. The red box highlights the search boxes you can use to filter 22
6 Example of how the bottom bar can be used to check how many changes we have made so far. 22

7 Example of how players are laid out in a map is automatic placement is used. 31
8 Connections generated by create_connect_all_players_land command. 39
9 Connections generated by create_connect_teams_lands command. Only connections between teammates are generated. 40

List of Tables

2.1 Allowed map sizes. 2

3.1 Examples of important paths used in Age of Empires II: DE . . . 5

1 Semantic of each Garrison Type value. 24

2 Macro to check if you want to detect the map size 46
3 Available seasons in Age of Empires II: DE. 48
4 List of all the default terrain types available 57
5 List of all the map types available 59

Acronyms

AI Artificial Intelligence. 29

DAT DATa. 19
DDS DirectDraw Surface. 9, 16, 17
DLL Dynamic Link Library. 19

EBNF Extended Backus–Naur Form. 26

PC Personal Computer. vii
PNG Portable Network Graphics. 17

RMS Random Map Scripting. i, ii, 13, 16, 26–59

UI User Interface. 16, 17
URL Uniform Resource Locator. 10

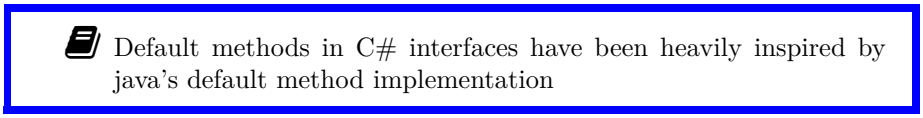
About this report

When writing this guide, the following convention are adopted.

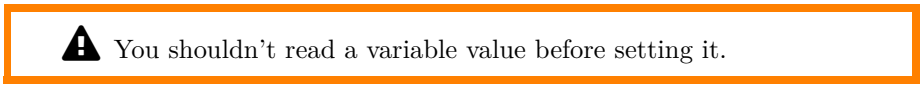
Generally speaking, definitions of concepts are shown as:

Definition 0.0.1. Square root of a number The square root of a number x ($\sqrt{x}$) is defined as the number that, multiplied with itself, yield x.

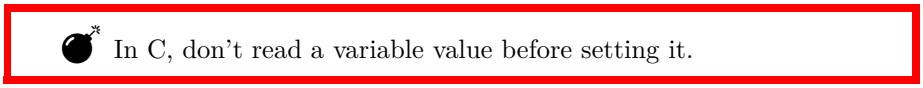
A note is something that adds context to a topic and is shown as below:



A warning is something that you should be aware of.



An attention contains information that, if not followed, will cause unexpected results;



Reference to the glossary are printed as below:
Personal Computer are used throughout the world.
Whole reference to the acronym table are shown as:
Personal Computer (PC) are used throughout the world.
Citation are shown as follows: in A* algorithm, we use $f = g + h$ to estimate search states [1].

Chapter 1

Introduction

This guide helps you creating a new mod in Age of Empires II: DE: by reading this file, you should be able to create simple mods. I have written this guide when I started my Age of Empires II: DE modding experience and contains all the information I have gathered while scouting the internet. I tried to include the most important information, but I have decided to leave some other (albeit relevant) information out from this document. Hence, the data presented here may be lacking or missing.

Note that this guide focuses its attention on Age of Empires II: DE, while other version of the game (e.g., Age of Conqueror or HD version) are explicitly not considered whatsoever: this usually means that some functions or characteristics may not be available in such game versions. The work here has been done in order to fix the stable bug from "Persistent Corpses" data mod [2] and in order to create more team maps where me and my friend can play in.

All code (alongside the source code of this document) is available at `https://github.com/Koldar/aoe-mod-tutorial`: please open an issue whenever you find any typos, missing references of false information.

Chapter 2

Background and Related Works

2.1 Maps

Definition 2.1.1. A map is said to be **nomad** when the players will start with no towncenter, and some villagers. Each player is required to create a town center wherever she desires.

Definition 2.1.2. A map is said to be **michi** the players will be separated by forest.

Cliffs (see Definition 2.1.3) are another element on the map. A cliff is shown in Figure 2.1

Definition 2.1.3. A cliff is an untraversable element of the map. Units on the cliff gain a strategic advantage w.r.t. unit under the cliff.

Maps have fixed **sizes**, which are shown in Table 2.1 [3].

Size	Tiles on Sides	Total tiles	Area ratio to 100x100 map
Tiny	120x120	14400	1.4
Small	144x144	20736	2.1
Medium	168x168	28224	2.8
Large	200x200	40000	4.0
Huge	220x220	48400	4.8
Gigantic	240x240	57600	5.8
Ludicruous	480x480	230400	23.0

Table 2.1: Allowed map sizes.

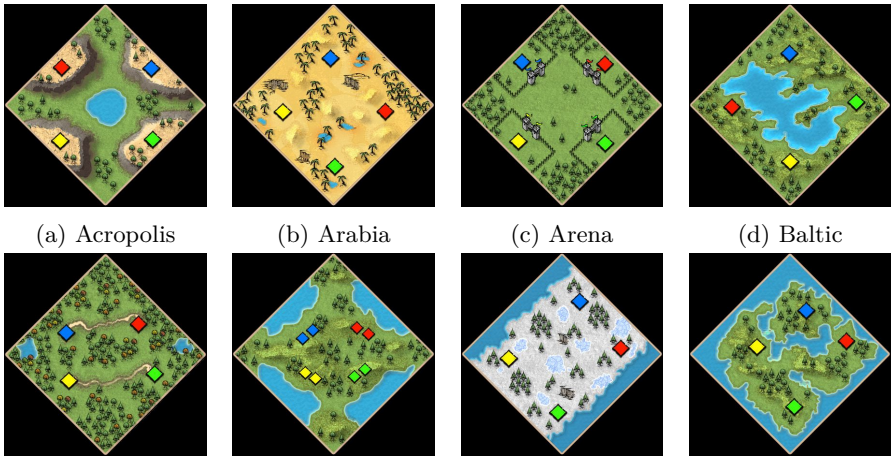
Figure 2.1: Example of a cliff in a map.

2.1.1 Types of maps

You can play Age of Empires II: DE on several different maps. A map can be played with a specific playstyle. The types of maps are the following ones:

- Random Maps
- Death Match;
- Battle Royale;
- King of the Hill;

Figure 2.2 shows some maps available to play in Age of Empires II: DE.



(a) Acropolis (b) Arabia (c) Arena (d) Baltic

(e) Black Forest (f) Budapest (g) Scandinavia (h) Continental

Figure 2.2: Examples of maps available in Age of Empires II: DE. Red, Yellow, Green and Blue diamond represents players positions. If a team game is played, Red and blue are allied against team yellow and green. A shoe represents a nomadic map.

Chapter 3

Basics and Definitions

The first thing is to install Age of Empires II: DE, for instance, via Steam [4]. After that, you need to identify Advance Genie Editor program: it is usually installed at "C:\Program Files (x86)\Steam\steamapps\common\AoE2DE\Tools_Builds\AdvancedGenieEditor3.exe". Data mods are discourages [5], hence you should create modes by replacing individual files and avoiding data modding whenever possible. We will refer to the folder `C:\Program Files (x86)\Steam\steamapps\common\AoE2DE\` as `Age of Empires II: DE Exe`.

The mods you install from Age of Empires II: DE are available in a directory that usually is `%HOMEPATH%\Games\Age of Empires 2 DE` (from here on dubbed `Age of Empires II: DE Home`). Within `Age of Empires II: DE Home`, there should be a folder whose name is the user **steam id** (which is a big number) (from here on such a folder is dubbed as `Age of Empires II: DE SteamId`) [2]. Within it, there is a folder called `mods\subscribed` containing all the mods the user has subscribed to. Along side "subscribed" folder, there is a folder named "local" where local mods in development may be put (such a folder will be referred to as `Age of Empires II: DE Local Mod`). The structure of `Age of Empires II: DE Home` is shown in Listing 3.1.

For example, Table 3.1 shows all the involved paths for the modding.

Path	Example
`Age of Empires II: DE Exe`	`C:\Program Files (x86)\Steam\steamapps\common\AoE2DE\`
`Age of Empires II: DE Home`	`C:\Users\FooBar\Games\Age of Empires 2 DE`
`Age of Empires II: DE SteamId`	`C:\Users\FooBar\Games\Age of Empires 2 DE\9823578647902347890`

Table 3.1: Examples of important paths used in Age of Empires II: DE

In the "mods" folder, there is a file called `mods-status.json` (an example of the content is shown in Listing 3.2).

```
1 [
2         {
3                 "CheckSum": "3716621926",
```

```
Age of Empires 2 DE
    |-- logs
        |-- ... logs ...
    |-- testharness
    |-- 0
    |-- 893465689237890
        |--mods
            |-- local
            |--subscribed
                |-- one folder per mod installed
            |--mod-status.json
```

Listing 3.1: Age of Empires home folder directory structure

```
4                "Enabled": true,
5                "LastUpdate": "1581833077032",
6                "Path": "subscribed//2695_Improved Small Trees",
7                "Priority": 1,
8                "PublishID": 0,
9                "Title": "2695_Improved Small Trees",
10               "WorkshopID": 2695
11       },
12       {
13               "CheckSum": "1216235570",
14               "Enabled": false,
15               "LastUpdate": "1575122745034",
16               "Path": "subscribed//1592_Better Main Menu Night Time",
17               "Priority": 2,
18               "PublishID": 0,
19               "Title": "Better Main Menu Night Time",
20               "WorkshopID": 1592
21       }
22 ]
```

Listing 3.2: Example of mods-status file

The json stored is a sequence of objects, each representing:

- Checksum: MD5 of the mod?
- **Enable**: if `true`, the mod is enabled, `false` otherwise;
- Last Update: timestamp representing the number of milliseconds when the mod has been lastly updated;
- **Path**: path, relative to `Age of Empires II: DE SteamId` directory, where the specific mod is installed;
- **Priority**: priority used to load the mod;
- Publish Id: always 0?

- **Title**: name of the mod to show to the user;
- WorkShop Id: An id that uniquely identifies this mod on `www.ageofempires.com` site;

At high level Age of Empires II: DE install each mod by following Algorithm 1. After sorting out the enabled mods, the software "patches" the set of files specified by the mod path. From the algorithm, it can be seen that the priority of each mod determine the order each mod is used. Priority may change the behavior is a mod x relies on the installation of the mod y (if $y.priority < x.priority$) [4].

Algorithm 1: Age of Empires II: DE mod boot strap algorithm

```
1 mods ← Gather mods in Age of Empires II: DE SteamId;
2 sortedmods ← Sort mods by prioritzing mods with small priority;
3 foreach m ∈ reversed(sortedmods) do mods subscribed
4 |   if ¬m.Enable then
5 |   |   continue;
6 |   end
7 |   Copy the files in Age of Empires II: DE SteamId\mods\m.Path
  |    into Age of Empires II: DE Exe;
8 end
9 Start Age of Empires II: DE;
```

Chapter 4

Creating a basic mod (MWE)

When developing a mod you should work in the directory `Age of Empires II: DE Local Mod`. Start by creating a folder with the same name as the mod you want to create. For this tutorial, we will create a mod that changes the background of the main menu: we will name such a folder "carcassone-menu".

For this very reason we need to put the background image (in this case the one shown in Figure 4.1)

Figure 4.1: Image to put as background

```
carcassonne-menu
    |-- widgetui
        |-- textures
            |-- backgrounds
                |-- mainmenu_bg.dds
    |-- info.json
    |-- thumbnail.jpg
```

Listing 4.1: Carcassonne mod layout

```
1 carcassonne-menu
2     |-- widgetui
3         |-- textures
4             |-- backgrounds
5                 |-- mainmenu_bg.dds
6     |-- info.json
7     |-- thumbnail.jpg
```

Listing 4.2: Carcassonne mod layout

As shown in the Appendix 6.0.1, the background image is specified (relative to `Age of Empires II: DE Exe`) in `widgetui/textures/backgrounds/mainmenu_bg.dds`. Hence, in the "carcassone-menu" folder, you need to create the subfolder `widgetui/textures/backgrounds/mainmenu_bg`; then you need to put the Carcassone image, name it "mainmenu_bg" (ensure that the image follows the DirectDraw Surface (DDS) extension).

Aside the folders you have just created, you need two additional files that represents mod metadata. Both files needs to be put in "carcassone-menu" folders. The first is `thumbnail.jpg`, which is a jpg format that is shown when browsing the mods. The other metadata file is call "info.json" (an example is shown in Listing 4.2): it is a json containing 3 values:

- Author: name of the author of this mod;
- Description: a description of the mod;
- Title: title to show of the mod;

If the developer puts other fields in this json, they will be automatically removed. Any formatting will be overwritten as well. All the metadata is shown in the right panel of the mod browsing window (in Age of Empires II: DE program, Mods section). Additional files in `Age of Empires II: DE Local Mod` mod will be left in the folder

The thumbnail can have any dimensions, although 400x150-ish dimensions may be preferred. After this operation, open Age of Empires II: DE, go to the mods, specifically to "My Mods". If you click "Import My Mods" Age of Empires II: DE will search into the `Age of Empires II: DE Local Mod` for compliant mods. You should see the new mods (as shown in Figure 4.2).

Figure 4.2: My mods window

The image in the background will be clipped. "mod-status.json" will be automatically updated.

> ⚠ If `thumbnail.jpg` is not present, a default image will be put instead. If `info.json` is absent, default values will be put: at the moment of writing, the author is set to "Unpublished", description to "No Description" and name as the same folder as the mod root dir. If "info.json" content is lacking one of those 3 fields, Age of Empires II: DE will automatically fill the missing ones.

In the mod window, if you select the "carcassonne-menu" you click "More Info". If you click this, Age of Empires II: DE will automatically open the browser at the Uniform Resource Locator (URL) `https://www.ageofempires.com/mods/details/XYZ` where XYZ is the WorkShop Id of the mod. If Age of Empires II: DE fails to load the mods, you can *debugging* the mod by make changes the mod in `Age of Empires II: DE Local Mod` and click "Import Local Mods" button.

4.1 Publish mod

It is now time to publish the mod. In this way the mod can be automatically downloaded by other client if it is required in a lobby game. In order to publish the mod you need to visit `www.ageofempires.com` and sign in with a *XBox Live Account* (Logging with the Steam one is not enough).

Choose the mod you want to publish and zip the content of the entire mod folder (i.e., the one containing the file "info.json" and "thumbnail.jpg"): this means that if you open the zip file, you will immediately find at the top level the files "info.json" and "thumbnail.jpg";[1] then, visit `https://www.ageofempires.com/mods/create/` and fill the form with all the required information and finally submit. After some time, you should have published your mode.

Alternatively, you can sign-in inside Age of Empires II: DE with you *XBox Live Account*. Then, you can visit te "My Mods" tab and click on "Publish Mod" or update when you want to publish a new version of the same mod.

[1] With `7-Zip`, if you right-click the zip and you "Open" it, you should find such files inside the windows that has just appeared.

Chapter 5

Implementing other mods using Advance Genie Editor

In this chapter we describe how to implement some more complex mods.

5.1 Altering corpses decaying

We now try to create a mod that simply changes the time when the corpse decays. Let us call it **ya-decaying-corpses**.[1] In this mod, we need to change data files. Data files are located in the `Age of Empires II: DE Exe\resources\_common`. As show in Appendix .10.5, you can use Advance Genie Editor to do so: Copy the file `Age of Empires II: DE Exe\resources\_common\dat\empires2_x2_p1.dat` in the mod dir; then open it via Advance Genie Editor. Switch to the *Units* tabs and select all the units of all type (except maybe Gaia's ones). For every one of them, go to the "Statisitcs" section and change the value *Resource Decay* to *-1*.

[1] "ya" stands for "Yet Another" and is a popular way of naming in computer engineering whenever the naming creativity is low.

Chapter 6

Creating Random Maps using RMS

In this chapter we will explain how to use RMS language to generate random maps. For information about RMS, see Appendix .13.3 or look at [6].

6.0.1 Filling PLAYER_SETUP

Appendices

Semantics of Age of Empires II: DE Exe

.1 BattleServer

Folder with `BattleServer.exe` file. Usually not useful when modding.

.2 certificates

`X509` file format of Age of Empires II: DE. Usually not useful when modding.

.3 Docs

Manual of PDF of Age of Empires II: DE. Not useful when modding.

.4 Schema

Not useful when modding.

.5 Support

Contains some link to Age of Empires II: DE website support. Not useful when modding.

.6 Tools_Builds

Represents Advance Genie Editor directory.

.7 webclient

Not useful when modding.

.8 wwise

Not useful when modding.

.9 resources

Specifies all the data that are not widgets of the User Interface (UI). The folder contains data like the cursors icons, the Advance Genie Editor dat files.

.9.1 _common

This is the main folder of **resources** directory.

cursors

List of all the cursors the cursor directed by the user mouse will display when a given action is performed. For instance, when you need to garrison a villager, a specific cursor image replace the classic arrow icon. All the cursors image have the `cur` extension.

dat

Contains the dat files containing all the information regarding units, civilization, buildings: `empires2_x2_p1.dat` contains all such information. [1]

drs

gamedata_x2 This important folder contains RMS files that can be used in your RMS programming (e.g., `F_Season.inc`). See Appendix .13.3.

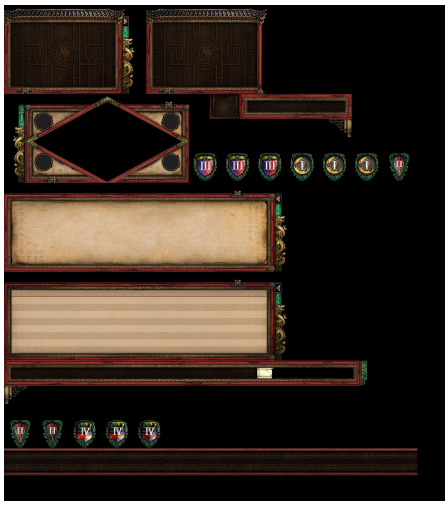
.10 widgetui

Specifies which set of textures Age of Empires II: DE needs to display alongside their configuration. If you need to alter a picture, it will probably be here. Contains a set of json. Furthermore, it contains a folder named "textures", representing all the textures and images in the game. There are versions, *textures* and *textures-sd*: the former contains the texture to use and the latter contains the low resolution version of the same textures. The first texture are mandatory while the second are optional (modwise). All textures are saved via DDS extensions.[2]

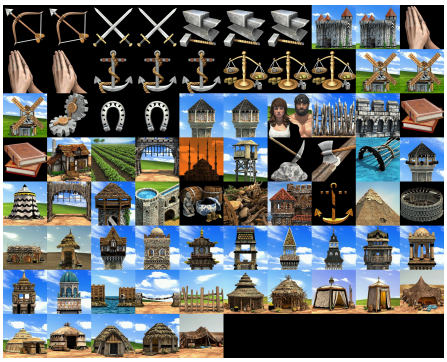
`Textures` folder contains other folders:

[1] The file can be opened by Advance Genie Editor

[2] You can use `Irfan View` 32-bit version to view DDS images.



(a) CivAsia file, containing the UI



(b) ingamebuildings file, containing the icons of the buildings in UI

Figure 2: Background menu examples

.10.1 atlas

Contains textures that are present in the Campaigns menu (e.g., the scenes where you can choose which campaign to play), Sun Tzu icons in the "Art of War", images representing the UI you see while playing (Figure 1a), the window chat, buildings icons (Figure 1b).

.10.2 backgrounds

Contains the menu background ("mainmenu_bg" and "mainmenu_bg_1") images as well the windows in the Age of Empires II: DE menus (e.g., lobby creation, history). Examples of these figures are shown in Figure **??**. Note that some of images here are represented via Portable Network Graphics (PNG) rather than DDS.

.10.3 campaign

Texture regarding the campaigns, one sub directory per campaign. For instance, "cam3" is the *Saladin Campaign*. Within it:

- *X.dss* file, where *X* is the campaign name, is the campaign image menu file (i.e., the image where you can choose the specific mission);
- *X_background* is the image where the mission intro and outro are presented;
- a subfolder, one per mission in the given campaign. Each sub folder name has a number (starting from 1) as name and contains the drawings in the intro and outros of the associated mission.

.10.4 ingame

TODO

.10.5 menu

TODO

Advance Genie Editor

This is a small guide to Advance Genie Editor software. The guide is based on version 2020.3.30. You should not consider this guide as something as official. Advance Genie Editor is a program for editing data of genie (DATa (DAT) and Dynamic Link Library (DLL)) files. It can edit properties of units, civilizations, technologies, graphics, terrains, sounds, player colors and some other things [7]. Advance Genie Editor program can be found in `Age of Empires II: DE Exe`, under "Tools_Build" directory.

When you execute it three windows automatically open. The *Open files...* allows you to open a Advance Genie Editor file. To manage Age of Empires II: DE files, click on the button *Age of Empires II: Definitive Edition* on the right of *Defaults:* label. Set the *Genie version:* to *Age of Empires II: Definitive Edition*. The file that you need to open is the one specified by *Compressed data set (*.dat)*: such a file should be called `empires2_x2_p1.dat`. When you open such a file you can view all the unit, buildings, civilizations parameters and properties. The vanilla file is `C:\Program Files (x86)\Steam\steamapps\common\AoE2DE\resources_common\dat\empires2_x2_p1.dat`.

After opening the file, you can now make changes on the file. Figure 3 shows the state fo the program when it opens the file.

As shown in Figure 4, Advance Genie Editor splits its parameters regarding the Age of Empires II: DE field.

> You can switch between tabs by the performing hotkeys `Ctrl + Pag. Up` and `Ctrl + Pag. Down`.

In the units left pane (called "Units") there is displayed the list of all the units a particular civilization can manage (e.g., in Figure 5 the civ is "Gaia"). Since the unit tabs show several parameters and unit, you can filter the units using the search box in the units pane (e.g., Figure 5 highlighted in the red box).

The search box works as follows: first you select the civilization, then you can add at most 2 criteria involving the unit name (in and with the civilization): namely, if the 2 criteria are γ_1 and γ_2 and the civilization involved is c, the whole search criterion is $civ = c \wedge (\gamma_1 \in unit.name) \wedge (\gamma_2 \in unit.name)$, where $unit.name$ is the string representing a generic unit name. For instance, if $\gamma_1 =$

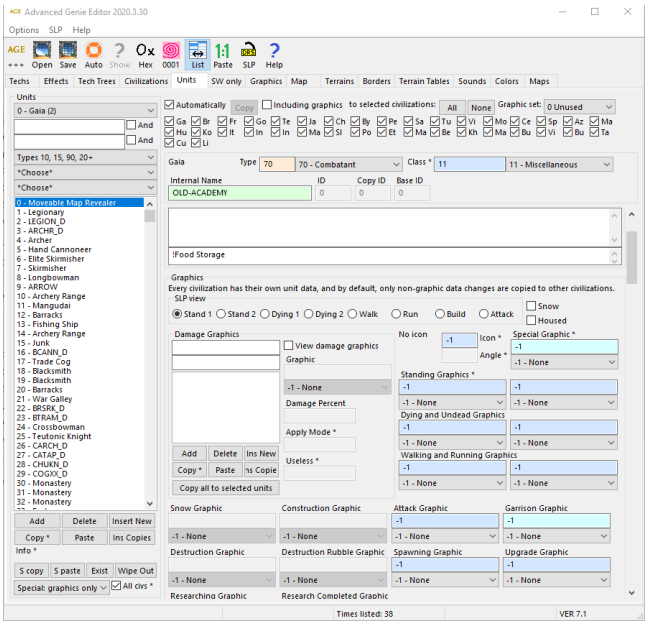


Figure 3: Standard Advance Genie Editor window

dead then we select all the units whose name contains the substring "dead". You can add "|" to perform a logical or ('∨') inside γ_i ($i \in \{1, 2\}$). For instance if $\gamma_1 = \_D|dead$, the global search criterion is $civ = c \wedge ((\_D \in unit.name) \vee (dead \in unit.name))$: such a search query will generate all the units whose name contains either "_D" or "dead".

You can select multiple units: when you alter a value from the unit pane, the same parameter will be updated also in every other civilization checked in the top checkboxes (e.g.,, "Ga", "Br", "Fr", "Go"). At the bottom you can see how many changes performed in this session (as shown in Figure 6).

.11 Required Knowledge

Unit names. The units ending with "_D" represent the corpses of the associated unit: for instance "Archer" unit is the actual unit with all its properties while "ARCHR_D" represents the archer corpse.

Buildings. Every time the user access to a new era, each building changes. For instance, in *graphics* you can see that there are, for each stable building civ, a stable for feudal, one for the castle age and another one for the imperial age.

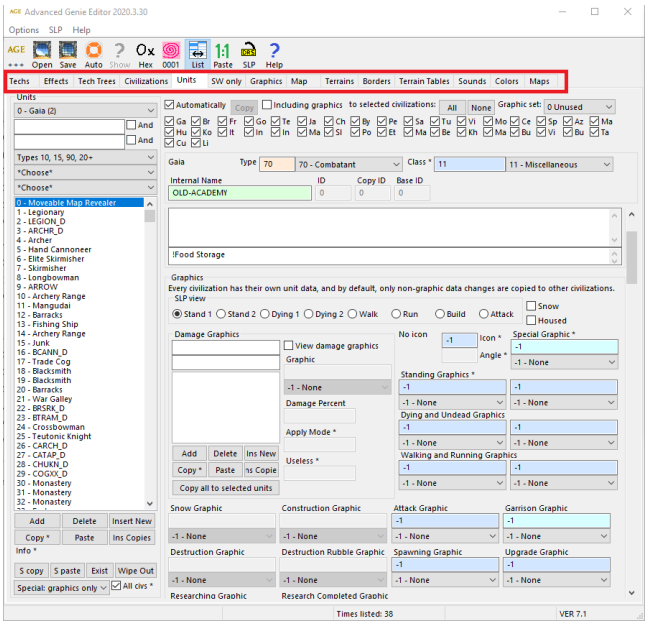


Figure 4: Standard Advance Genie Editor window. The red box highlights the category tabs grouping the parameters you can tweak according to the Age of Empires II: DE field.

.12 Units parameters

Manage a single unit parameters. This section is divided in multiple sub areas:

- Language Files;
- Graphics;
- Statistics;
- Projectiles;
- Attributes;
- Sounds;
- Tasks;

The documentation in this area has been copied from [8].

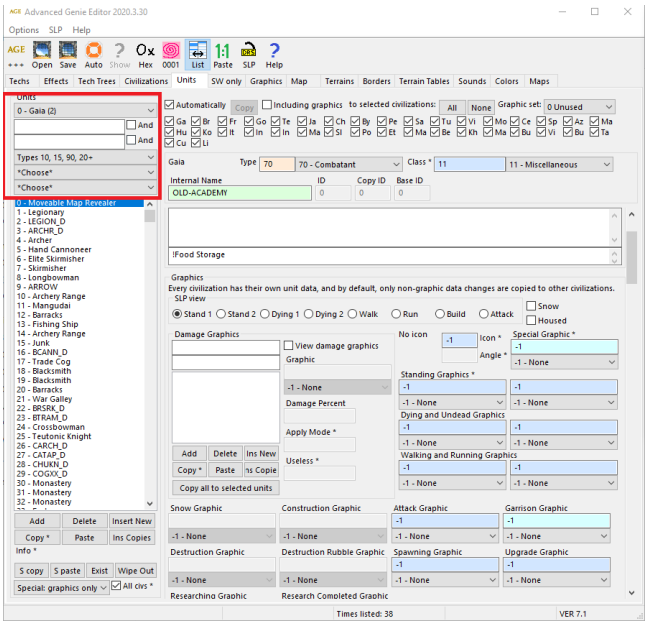


Figure 5: Standard Advance Genie Editor window. The red box highlights the search boxes you can use to filter

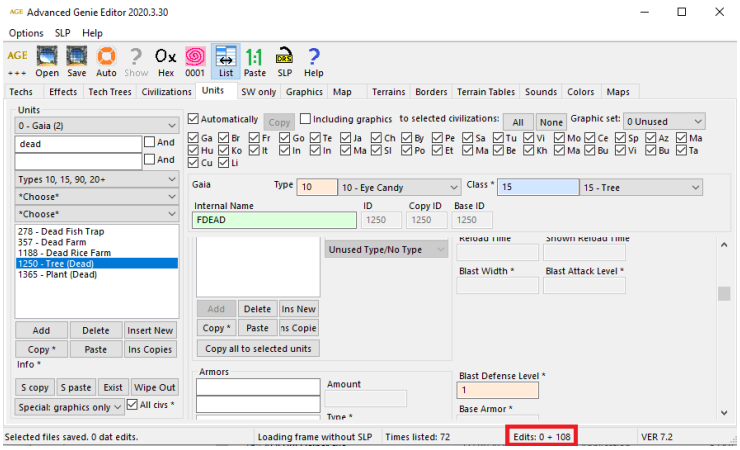


Figure 6: Example of how the bottom bar can be used to check how many changes we have made so far.

.12.1 Language Files

.12.2 Graphics

.12.3 Statistics

Hit Points: hit points of the unit. If -1, the unit will immediately die [9]. **Speed**: the speed of the unit. **Rotation Speed** makes the unit slower. **Line of Sight** represents the number of cells the unit can see. Generates a circle centered in the unit. **Search Radius** is determines the area within which the unit will recognizes and react to other units. If anything enters this area, the unit will notice it; If you enter a value higher than LOS, a unit will chase enemy units outside of its sight range. **Min Range** (**max Range**) represents the minimum (maximum) range the ranged unit can shoot its projectile: for example an archer can shoot any target between 0 and 4 cells [8].

Resource capacity determines the quantity of resources a unit can hold (in range $[0, 3392]$): for villagers, this is the max amount of resources they can carry. For other units, it can be other resources, like the time it takes for a dead unit to decompose [8].

Resource Decay determines how fast objects (like corpses) decay. Set -1 for never decaying. A negative value will prevent the corpse from decaying permanently. The larger a positive value is, the longer the corpse will last. (The value is approximately the equivalent number of game seconds). Many non-corpse units have resource decay value, but the purpose of it in other units is unknown [8].

Work Rate determines the base work speed of units. This affects rates like villager work rates (building, gathering, etc.), conversion speed, and trebuchet pack (unpack) speed. the value needs to satisfy the constraint $x \geq 0$ [8].

Garrison Capacity is the number of units a transport or building can hold. allowed value x must satisfy the constraint $x \in [-1, 127]$. Only rams, buildings, and transport ships can normally hold units. [3] If you give a rams an original speed of 0, garrisoning units will allow it to move.

Garrison type determines what units can garrison into the building ($x \in [0, 15]$). Such a integer value should be interpreted as an enumeration, which semantic is shown in Table 1. The number are encoded in Advance Genie Editor as a 8-bit value [8].

Garrison Heal Rate determines how fast a building heals units garrisoned within it ($x \geq 0$). This is different from the Garrison Recovery, which determines how fast the unit heals when garrisoned in any building [8].

.12.4 Projectiles

.12.5 Attributes

Dead Unit Represents the unit representing the corpse of the unit currently under analysis. By convention, all "dead units" ends with `_D`. If a unit does not

[3] Garrisoning infantry into rams will increase both the ram's speed and attack.

Value	Semantic
0	None
1	Villagers only
2	Infantry only
3	Villagers and infantry
4	Cavalry
5	Cavalry and villagers
6	Cavalry and infantry
7	Cavalry, infantry and villagers
8	Monks only
9	Monks and villagers
10	Monks and infantry
11	Monks, infantry and villagers
12	Monks and cavalry
13	Monks, villagers and cavalry
14	Monks, cavalry and infantry
15	Monks, villagers, infantry and cavalry

Table 1: Semantic of each Garrison Type value.

have any dead unit, this field is set to -1. For example, the dead unit of the unit "Archer" is set to be `ARCHR_D` unit.

.12.6 Sounds

.12.7 Tasks

.13 Graphics

Represents the graphics and the animations involved in every units and buildings. This section has been copied from [10]. The left pane represents the sprites of each unit, building and so on. There is a search box that is similar to the one in "Units".

Internal Name is the name used to identify the sprites in the Advance Genie Editor in the left pane of "Graphics" panel.

Frames per Angle represents the framerate of the sprite.

Animation Duration represents how long the animation is displayed in seconds. After that time the animation stops and the sprite is removed. It is required that $x > 0$. Exponential notation can be used (e.g., $e3 + 35$). Use this field if you want to avoid the sinking of dead units for every sprite whose name contains "(Decay)" substring.

.13.1 Deltas

Graphic represents the id of the sprite to show. The name of the file refers to the json file present in `resources/_common/particles`.

.13.2 Deltas

.13.3 Angle Sounds

RMS file format and layout

RMS files are used to generate random maps that follows some specific pattern. Random maps change each time you play since they are randomly generated starting from a seed s. Examples of such maps are *Black Forest* or *Scandinavia*. If you need to create a new type of random map, you need to provide a `rms` file. `rms` files are written by using a language (called *rms*) created by the developers of Age of Empires II: DE [11]. This section will describes you a RMS file is structured. This section is based on the guide made by Ryan Andrews [11] and by [6].

At high level, a RMS file has 7 basic sections:

1. PLAYER_SETUP;
2. LAND_GENERATION;
3. ELEVATION_GENERATION;
4. CLIFF_GENERATION;
5. TERRAIN_GENERATION;
6. CONNECTION_GENERATION;
7. OBJECTS_GENERATION;

As an example, consider Listing 1 made by vierklee, which represents a random map which create an area where all the teams are located behind procedurally generated walls.

At high level, in a `rms` file comments follows `C` conventions (i.e., comments start with "/* " and end with " */"). Each section starts with a **section declaration** and is followed by a section body. Each section declaration follow the regular expression `<[a-zA-Z0-9_]+>\n`. Listing 2 specifies the Extended Backus–Naur Form (EBNF) of the RMS language. The language is case-sensitive and indendation is ignored [6].

```
Age of Empires 2 DE
    |-- logs
        |-- ... logs ...
    |-- testharness
    |-- 0
    |-- 893465689237890
        |--mods
            |-- local
            |--subscribed
                |-- one folder per mod installed
            |--mod-status.json
```

Listing 1: The full source code of the rms file which generates the team maps "team arena".

.14 RMS control flow

.14.1 Conditional flow

In RMS conditional flow (i.e., `if`, `then`, `else`) are implemented via the keywords `if`, `elseif`, `else`, `endif`. Both `elseif` and `else` are optiional. Listing .14.1 shows an example of how to use the conditional flow.

```
        if FOOBAR
            /* do something when FOOBAR preprocessor directive is
                ↪ specified */
        elseif FOOBAZ
            /* do something when FOOBAZ preprocessor directive is
                ↪ specified */
        else
            /* do something else */
        endif
```

.14.2 #include_drs

Include a file located in a given folder. If the filepath is relative, it is relative to the folder `Age of Empires II: DE Exe\resources\_common\drs\gamedata_x2` (e.g., `C:\Program Files (x86)\Steam\steamapps\common\AoE2DE\resources\_common\drs\gamedata_x2`). Inside such a folder, you can several include files that you can use to ease your rms programming. The content of the included file will be dumped as-is in the current file.

.14.3 #const

`C`-like preprocessor. Every time the application see the identifier, it will be replaced by the associated value.

```
        #const ANSWER 42
```

For example, in Listing .14.3 the statement allows the program to replace `ANSWER` with the string 42.

```
identifier <- [a-zA-Z_]([a-zA-Z0-9_])*
string <- [a-zA-Z_\.\-0-9]+
number <- [0-9]+

rms-file <-
    section*


section <-
    '<' identifier '>' section-statement*

section-statement <-
    preprocessor-statement
    | command-statement
    | command-complex-value-statement
    | if-statement

preprocessor-statement <-
    "#define" identifier
    | "#include_drs" string
    | "#include" string

command-statement <-
    identifier value*

command-complex-value-statement <-
    identifier '{' section-statement* '}'

if-statement <- "if" identifier section-statement+ elseif-statement*
    ↪ else-statement? "endif"

elseif-statement <- "elseif" identifier section-statement+

else-statement <- "else" section-statement+

value <-
    number
    | string
```

Listing 2: EBNF-like statements representing the language of the rms file.

.15 RMS file sections

In the following, we will describe each section.

.15.1 PLAYER_SETUP

Player placement In `PLAYER_SETUP` we need to first choose how the players starting position are put in the map. So, you need to put one of the following specifications:

- **random_placement**: players are positioned in a circle/oval. This is the default value;

- **grouped_by_team**: players are positioned in close proximity to each other. Distance between team members is double the base_size used in create_plater_lands;
- **direct_placement**: Allows to manually set the player positions. via create_land attribute in LAND_GENERATION. For example in Listing **??** the first player is positioned in the middle of the map.

```
1                <PLAYER_SETUP>
2                direct_placement
3                <LAND_GENERATION>
4                create_land {
5                        terrain_type DESERT
6                        land_percent 3
7                        land_position 50 50
8                        assign_to_player 1
9                }
```

If you plan to make the map as a nomad map, you should set the command nomad_resources: this will add the cost fo a town center to each player.

ai_info_map_type Next thing you can set is the Artificial Intelligence (AI) that Age of Empires II: DE will use in the map. You can see as a hint that Age of Empires II: DE will use. Such an hint is declared via command ai_info_map_type. The signature of the command is the one shown in Listing .15.1, where:

```
1        ai_info_map_type MapType IsNomad IsMIchi ShowType
```

- MapType is the map type constants (see Section .19.2);
- IsNomad either 1 or 0: if you plan to make the map as "nomad", you are **required** to set this value to 1;
- IsMichi: either 1 or 0: if true a forest will **completely** separating the players;
- ShowType: either 1 or 0: if true, the map type will be shown in the objective tab;

.15.2 LAND_GENERATION

Place some large areas of terrain in the map. For instance, some maps have some texture representing ground at each player starting location while the rest of the map is filled with grass. This section of the file performs this very feature.

The first to declare in this section is the type of the terrain you want to have, which can be achieved by using base_terrain command. Such a command requires a single input, which should be fetched from the terrain types (see Appendix .19.1).

Now you need to create large pieces of terrains on the maps. Terrain maps can be used to create a walkable texture at the starting position of a player or a lake in the center like the baltic map [3].

create_player_lands

The command allows you to create a specific **land** terrain for **every** player in the map. This command applies to all the players present in the map. If you want to assign different terrains to different players, consider using create_land with assign_to_player instead. The command is optional in the script. If the command can be used multiple times, but every player will have multiple town centers as well as result: this is probably how map "Budapest" is coded. Note that you give to a player any land, you cannot give to her any starting units or resources as well, so it is **very important** that each player has at least one starting land.

create_player_lands cannot be used, due to a bug, alongside direct_placement.

Listing 3 shows an example of usage of the command. It will create, for each player, some mass of initial land for the players, where [12]:

```
1   create_player_lands
2   {
3       terrain_type                    GRASS
4       base_size                       19
5       land_percent                        5
6       clumping_factor                 3
7       border_fuzziness                15
8       set_zone_by_team
9       other_zone_avoidance_distance   0
10      circle_placement
11      circle_radius 38
12  }
```

Listing 3: Example of usage of create_player_lands in Team Arean by vierklee.

Figure 7 shows how teamless map generates each player starting position base.

The command requires several parameters, which are described next.

terrain_type the terrain you want to generate. See Section .19.1.

land_percent Percentage of the total map that the land to create should cover. the number belongs to the range $[0, 100]$ (default to 100). If used in create_player_lands, the percentage is divided equally by all the players.

Figure 7: Example of how players are laid out in a map is automatic placement is used.

number_of_tiles Fixed number of tiles that the land should grow by, in addition to the land specified by base_size. When behavior_version is set to 1, the square origin is included in the total number of tiles, resulting in smaller lands. If used in create_player_lands, the player share the same amount of number of tiles.

land_position Allows you to specify the exact origin point for a land, as a percentage of the total map dimensions. It requires 2 parameters x and y which are, respectively, the x coordinate from the left point to the right and the y coordinate from the bottom to the top of the starting point. You need to satisfy the constraint $x \in [0, 100]$, $y \in [0, 99]$. The command cannot be specified when added into create_player_lands or when assign_to_player or assign_to is specified into create_land (unless direct_placement is specified in PLAYER_SETUP). The command ignores border restrictions and, if the land is placed outside the border, it will not grow beyond its base_size.

left_border, right_border, top_border, bottom_border All needs to satisfy the constraint $x \in [0, 97]$, default to 0. They specify an subset of the map where the land will be put. For instance, this allows the developer to put the players only in the bottom right part of the map (e.g., in Pilgrims); in create_player_lands, the border coordinates effectively shift the circle where the players are put; when using this function, the created land will have octoagonal shape. For further information, see `https://docs.google.com/document/d/e/2PACX-1vR_I1I9u-hI2FFm-EYyjoM_-9dNJFOfTaIgr05wXNbdpv9tI18b6r18ARULy3Jqyvlq6-lc2VjX6xP4/pub#h.xrncn5cs75or`.

set_zone_by_team

base_size It is the radius (in tiles) that each player starting position land will have, in map cells; The bigger it is, the bigger is each player base.

circle_radius It represents the radius of the circle where each player base will be placed. The bigger it is, the more distant each player will be w.r.t.the other ones. The players are put on the circonference of said circle; You need to specify 2 numbers a and b: if $a \neq b$, it is produced an ellipse instead of a circle; If $a = b$, b may not be specified.

circle_placement

border_fuzziness It applies some noise to each player starting land borders ($x \in [0, 100]$). Values near 0 corresponds to perfectly rectangular bases while bigger values tends to create bases with a square-less shape. A good value may be 15; border_fuzziness Specifies the extent to which land growth respects the assigned borders. Bigger values means that the borders are fully respected while values near 0 means that the generated area may go beyond the specified border.

clumping_factor Alters the shape of the created land ($x \in [0, 100]$). High values represent rounder lands while low values represents more elongated shapes [6, 13].

base_elevation elevates the entire land by the specified height ($x \in [0, 9]$). The value is ignore if the terrain is water. Slopes are automatically added to blend the elevation with the sorrounding terrain.

assign_to_player give ownership of this land to the given player. Allowed values are $x \in [1, 8]$ (which are the player lobby positional number). If missing, the land is neutral. Lands assigned to players which are not playing will not be created. Mutually exclusive with assign_to.

assign_to Given ownership of this land to the given entity. Mutually exclusive with assign_to_player. It requires 4 additional arguments.

1. AssignTarget, which tells which entity will take ownership of the land. Can be either AT_PLAYER, AT_COLOR or AT_TEAM;

2. Number x: represents the specific entity Id of AssignTarget $t \in 0, 1, 2$. If t = AT-PLAYER (0), $x \in [1, 8]$ (refers to the lobby order); if t = AT-COLOR (1), $x \in [1, 8]$ (refers to the player color); If t = AT-TEAM (2), $x \in \{-10, -4, -3-, -2, -1, 0, 1, 2, 3, 4\}$ (refers to the lobby order of the team, not the number chosen by the team): in this cse is the team 0 refers to unteamed players, while any negative value x represents a player

that is not in the team whose lobby order is $-x$. Team -10 is a special team that consists to any player;

3. Mode m: semanticful only if AssignTarget si set to AT_TEAM. either -1 or 0. 0 means the land is assigned randomly in the team while -1 means that the land is assigned in an orderly matter;

4. Flag $f \in [0, 3]$. It is a 2-bit biset number. First bit determines if we need to reset players who have already been assigned before startign whiel the second bit tells to avoid remembering the fact that we have assign to a given player this land;

When you assign a piece of land to a player, you can also place object owned by such a player. Lands owned by players which are not playing are not generated at all.

zone Allows the land you are creating to be assigned with a particular **zone id**. Lands which share the same zone id are allwoed to touch eacother when growing. If 2 lands that are going to overlap have different zone ids, will be forbidden to do so: even better, they are garantueed to be distant by other_zone_avoidance_distance. The command is mutually exclusive with set_zone_by_team and with set_zone_randomly. Lands created with create_player_lands have their own unique zone id, while all lands created with create_land share the same zone id.


Zone 99 will crash the game.

set_zone_by_team Assign the same zone to all the members of the same team. Usage is recommended only in create_player_lands.

set_zone_randomly TODO

other_zone_avoidance_distance Number of cells that each starting player position needs be distant from the other zones. If such a value is set to 0, each starting position base may overlap with others (like in Team Arena by vierklee).

min_placement_distance

land_id Assign a numeric label to a land, which can later be used to place object specifically on that land via place_on_specific_land_id command. Multiple lands may share the same ID. The commands needs to be used after assign_to_player or assign_to, since they will reset the ID.

create_land Creates a single piece of land which is owned by no players. Hence it can be used to place neutral mass of lands (e.g., shared continent or shared center lakes, like in Baltic). As an example, consider Listing 4: the example create a lake centered in the middle of the map. The lake itself covers 20% of the map

```
1        create_land
2        {
3            terrain_type WATER
4            land_percent 20
5            land_position 50 50
6        }
```

Listing 4: Example of usage of create_land in Team Arean by vierklee.

.15.3 CLIFF_GENERATION

Thanks to this section, cliffs are added in the map [14, 3]. Cliffs are generated *after* lands but *before* terrain. If the section declaration is left empty, some cliff are still generated on the map. To further customize the cliffs, you need to input other commands.

min_number_of_cliffs The mininum number of cliffs to generate. It requires the number of minimum cliffs to generate. For instance, in Listing **??** we generate at least 10 cliffs.

```
<CLIFF_GENERATION>
min_number_of_cliffs 10
```


If you put a large number of cliffs, the number of cliffs will be truncated if there is not enough space in the map.

max_number_of_cliffs The maximum number of cliffs to generate. It requires the number of maximum cliffs to generate. For instance, in Listing **??** we generate at most 10 cliffs.

```
<CLIFF_GENERATION>
max_number_of_cliffs 10
```

min_distance_cliffs The minimum number of **cliff unit** that separates one cliff to another one. It accepts a single number $x \geq 0$, whose default value is 2.

Definition .15.1. A cliff unit corresponds to 3 map tiles.

min_length_of_cliff and max_length_of_cliff The minimum (and the maximum) length of a generic cliff. Note that you are required to satisfy $min \geq max, min \geq 0, max \geq 0$.

cliff_curliness Determines how much the cliffs tends to change directions ($x \in [0, 100]$). At 0, cliffs will be very straight and at 100 they will be extremely twisted. For example in Listing **??** we will generate exactly 20 cliffs, which are a bit "curly".

```
<CLIFF_GENERATION>
min_number_of_cliffs 20
max_number_of_cliffs 20
cliff_curliness 10
```

.15.4 TERRAIN_GENERATION

Replace terrains with other terrains. It is usually used to place clumps of forest, adding textures to make the map more nice, adding small lakes into chunks of forests and so on. The main command you should use here is create_terrain, although you can also use the following commands:

- color_correction;

color_correction

Specify a color correction to the whole map. Within the game, such an option is available via "Map Lightning". The command accepts a single argument x, which can be one of the following: `CC_AUTUMN`, `CC_WINTER`, `CC_JUNGLE`, `CC_DESERT`.

create_terrain TERRAIN

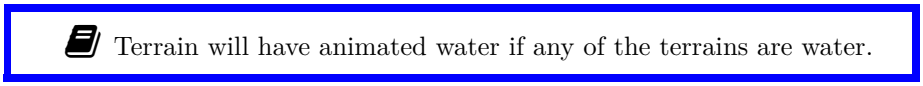
Create a single piece of terrain. Depending on which terrain you add in the map, the behavior may change. It accepts a lot of arguments, hence we will deal with them in several paragraphs.

base_terrain TERRAIN It performs like the argument in the land generation.


Due to a bug, the value does not default to GRASS.

terrain_mask NUM Enables terramin masking or layering for the terrain being created. The terrain to add inherits all properties, placement restrictions, automatic objects (e.g., trees for forest terrains or minimap color) from the base terrain. This function is great to blend terrains in order to create more realistic

and aesthetically pleasant maps. The only argument is the layer number, which can either 0 (no masking), 1 or 2. Setting the layer to 1 means that the new terrain is masked *above* the base terrain ajnd inherits its properties. On the other hand, setting the layer to 2 means that the terrain to ad is placed *below* the base terrain: in this case the base terrain will inherits the properties from the terrain just added.



spacing_to_other_terrain_types NUM Minimum distance that this terrain will stay away from other terrain types. It requires a number $x \geq 0$ (default to 0). This value considers only the terrains already present. The command will ignore terrain sharing the same terrain type. If set_flat_terrain_only, the terrain will distance that the terrain will stay away from slopes.

set_flat_terrain_only Command the map generation to avoid slopes tiles by the distance specified by the only argument x, $x \geq 1$. Listing .15.4 shows an example where the generator creates a hill where the bottom and the top are desert, but the slopes are grass.

```
1        <ELEVATION_GENERATION>
2        create_elevation 7 {
3            base_terrain GRASS
4            number_of_tiles 3000
5            number_of_clumps 1
6        }
7
8        <TERRAIN_GENERATION>
9        create_terrain DESERT {
10           base_terrain GRASS
11           land_percent 100
12           number_of_clumps 9320
13           spacing_to_other_terrain_types 1
14           set_flat_terrain_only
15       }
```

land_percent NUM Percentage of the total map allocated to this create_terrain command. If number_of_clumps is specified, this value is divided equally among the clumps. Listing .15.4 contains an example where a a desert covers 50% of the map. The terrain will only be replaced where the appropriate base_terrain or base_layer is present, and will only replace the specified number of individual clumps, so it will not necessarily fill 100% of the map if set to 100.

```
1        <TERRAIN_GENERATION>
2        create_terrain DESERT {
3            base_terrain GRASS
4            land_percent 50
5        }
```

number_of_tiles NUM Total time count allocate dto this create_terrain command. If number_of_clumps is specified, this value is divided equally among the clumps.

number_of_clumps NUM TODO

clumping_factor NUM TODO

set_scale_by_size Scales number_of_tiles to the map size. Unscaled value refers to a 100x100 map (see: Table 2.1). If you see a script scaling by both size and groups, only the final attribute will apply! If you want to scale by both groups and size, use set_scale_by_groups instead. Listing .15.4 contains an example where the map generator creates 4 lakes which become larger on larger maps; on a small map this will be 4 clumps with a total of $400 \cdot 2.1 = 840$ tiles. The command is mutually exclusive with set_scale_by_groups.

```
1        <TERRAIN_GENERATION>
2        create_terrain WATER {
3            base_terrain GRASS
4            number_of_clumps 4
5            number_of_tiles 400
6            set_scale_by_size
7        }
```

set_scale_by_groups Scales number_of_clumps to the map size. Unscaled value refers to a 100x100 map (see Table 2.1). If you see a script scaling by both size and groups, only the final attribute will apply! When used with number_of_tiles, the total tiles are also scaled to map size as well (but only for terrains, not for elevation). For example, Listing .15.4 allows to Create 400 tiles worth of lakes, with the number of lakes *and* the total number of tiles scaling to map size. On a small map this will be 4x2.1 = 8 clumps with a total of $400 \cdot 2.1 = 840$ tiles.

```
1        <TERRAIN_GENERATION>
2        create_terrain WATER {
3            base_terrain GRASS
4            number_of_clumps 4
5            number_of_tiles 400
6            set_scale_by_groups
7        }
```

set_avoid_player_start_areas

height_limits NUM NUM The terrain will only be placed on tiles of height between min and max (inclusive). For most purposes, values between 0 and 7 are useful, where 0 being the standard non-elevated height and 7 being the max height that can be produced by create_elevation. For example Listing .15.4 allows you to create a hill and place desert terrain only on the slopes.

```
1         <ELEVATION_GENERATION>
2             create_elevation 7 {
3             base_terrain GRASS
4             number_of_tiles 3000
5             number_of_clumps 1
6         }
7
8         <TERRAIN_GENERATION>
9             create_terrain DESERT {
10            base_terrain GRASS
11            land_percent 100
12            number_of_clumps 9320
13            height_limits 1 6
14        }
```

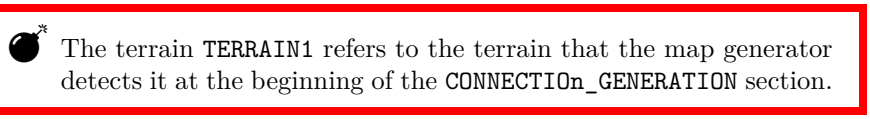
.15.5 CONNECTION_GENERATION

The section allows to replace terrains with other terrains, specifically along a given pah between the origins of lands. Used to create roads between players, shallows area across rivers and to ensure that forests do not completely separate players. If a connection provided by the RMS file is not possible, it is skipped. Each commands creates several connections between points and perform an action over the terrains.

Each command can be associated to several attributes, which are described below:

default_terrain_replacement TERRAIN Replace all the terrains in the connections with the specified terrain.

replace_terrain TERRAIN1 TERRAIN2 If the terrain `TERRAIN1` is part of a connection, replace it with the terrain `TERRAIN2`. Can be used multiple times for different terrains.



terrain_cost TERRAIN NUM The cost of having the connection run through the specified terrain. This allows the path finding algorithm [15] to alter the optimal solution. If the terrain is the one `TERRAIN`, the cost of such edges are temporary set to `NUM` instead of 1. $NUM > 0$. Listing .15.5 shows an example of how to use this attribute.

```
1         <CONNECTION_GENERATION>
2         create_connect_all_players_land {
3             replace_terrain GRASS ROAD
4             replace_terrain FOREST LEAVES
5             replace_terrain WATER SHALLOW
6             replace_terrain MED_WATER SHALLOW
```

```
            replace_terrain DEEP_WATER SHALLOW
            terrain_cost GRASS 1
            terrain_cost FOREST 7
            terrain_cost WATER 7
            terrain_cost MED_WATER 12
            terrain_cost DEEP_WATER 15
        }
```

terrain_size TERRAIN NUM NUM When a connection passes through a tile of the specified terrain, the area within radius +/- variance will be subject to replace_terrain or default_terrain_replacement and terrains will be replaced accordingly. Listing .15.5 shows an example of how to use this attribute: the roads are slightly larger. The first argument represents the radius of the road we are building while the second parameter represents the variance of the radius of the road we are building.

```
        <CONNECTION_GENERATION>
        create_connect_all_players_land {
            replace_terrain GRASS ROAD
            replace_terrain WATER SHALLOW
            terrain_size GRASS 1 1
            terrain_size WATER 3 1
        }
```

create_connect_all_players_land

Create several connection in order to make sure that all lands of players (both the starting ones and the ones assigned to them) are reachable. Each connection edge (u, v), where u, v represent two players. An example of the connection generated are shown in Figure 8.

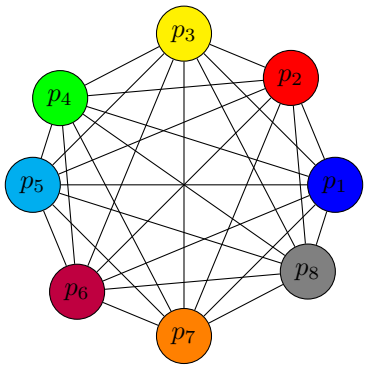


Figure 8: Connections generated by create_connect_all_players_land command.

As an example of how to use this command, you see Listing 5.

```
        <LAND_GENERATION>
        create_player_lands {
            terrain_type DESERT
            number_of_tiles 100
        }

        <CONNECTION_GENERATION>
            create_connect_all_players_land {
            default_terrain_replacement DIRT
        }
```

Listing 5: Example showing how you can connect all players with dirt.

create_connect_teams_lands

Create several connections. Each connection edge (u, v), where u, v represent two players starting position belonging to the same team. An example of the connection generated are shown in Figure 9. The road generated may go into enemy teams or neutral teams as well, if the connection is the optimal one.

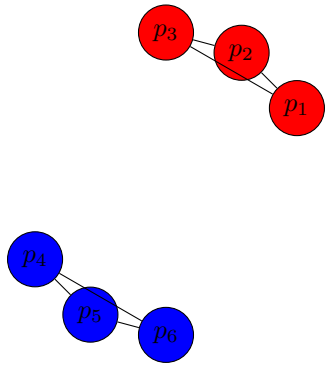


Figure 9: Connections generated by create_connect_teams_lands command. Only connections between teammates are generated.

Listing 6 shows an example where the command is used to replace the terrain to ROAD.

```
        <LAND_GENERATION>
        create_player_lands {
            terrain_type DESERT
            number_of_tiles 100
        }

        <CONNECTION_GENERATION>
        create_connect_all_players_land {
            replace_terrain FOREST ROAD
            replace_terrain GRASS ROAD
        }
```

Listing 6: Example showing how you can connect teammates with roads.

create_connect_all_lands

Connections will be generated in order to connect all the lands present on the map.

create_connect_same_land_zones

Apparently the same of create_connect_all_lands.

create_connect_to_nonplayer_land

Connect all player lands to all neutral lands, but do not directly generate connections between individual players.


After calling this command, all other connection commands appearing afterwards will not work anymore.

.15.6 OBJECTS_GENERATION

This section deals when placing buildings, units, resources, animals, straggler trees, decorations and other stuffs. Objects are placed in the same order as they appear in the RMS file. Normally, each tile can contains at most 1 object. If the object cannot be placed, it is not generated at all. The command that places objects is performed via create_object command.


Consider placing first the most important objects.

The attributes of create_object are shown below (since they are many).

number_of_objects NUM Number of objects to places in the whole map. The num needs to $x \in [1, 9320]$. Listing 7 allows to generate 10 batches of gold mines in the whole map.

```
1       <OBJECTS_GENERATION>
2       create_object GOLD {
3           number_of_objects 10
4       }
```

Listing 7: Example showing how you can create object in the whole map.

number_of_groups NUM Place a number of objects grouped together. Note that the total number of objects added is $n \cdot g$, where n is number_of_objects and g is the number_of_groups. Listing 8 allows to generate 20 groups of gold mines, each with 5 patches of gold.

```
<OBJECTS_GENERATION>
create_object GOLD {
    number_of_objects 5
    number_of_groups 20
}
```

Listing 8: Example showing how you can create object in the whole map, grouped together.

groups_variance NUM The number_of_objects is randomly varied up to ± the amount specified. Note that this is not the mathematically variance, but the standard deviation. Each group generated has a different number of objects. Listing 9 allows to generate 10 groups of gold mines, each with patches of 5 ± 3 gold.

```
<OBJECTS_GENERATION>
create_object GOLD {
    number_of_objects 5
    number_of_groups 10
    group_variance 3
    set_tight_grouping
}
```

Listing 9: Example showing how you can create different groups of object in the whole map, with different number of objects in each group.

resource_delta NUM TODO

second_object OBJECT TODO

set_scaling_to_map_size TODO

set_scaling_to_player_number TODO

set_place_for_every_player Place the object(s) as a personal object for each player (actually for each player land). Objects that cannot be owned by players (e.g., boar, gold, trees, and so on) also require set_gaia_object_only to be placed for every player.

Only works for player lands or lands assigned to players. Disabled by land_id. Objects will only be placed where they are not separated from the origin of their land by a terrain they are restricted on.

This means that on islands your resources will only end up on your own island. It also means that player gold mines on acropolis can only be placed on the hilltops, because gold is restricted from the rock terrain. Water objects (docks/boats) CAN be placed if the player land is made of a dirt terrain type. Terrain restrictions can be bypassed by using a placeholder and second_object.

Even though road terrains are restricted for resources, they do not form a separation the way other terrains do. Listing 10 allows to give every player their starting villagers.

```
        create_object VILLAGER {
            set_place_for_every_player
            min_distance_to_players 6
            max_distance_to_players 7
        }
```

Listing 10: Example showing how you can give to each player a single villager.

place_on_specific_land_id NUM TODO

set_gaia_object_only Use it alongside set_place_for_every_player to place gaia objects on a per-player basis. Muyst be used when placing neutral units (e.g., stone, deer, boars). Units and buildings will permanently join the player who first finds them unless set_gaia_unconvertible is specified. Gaia building architecture (in case of buildings) can be specified via set_gaia_civilization.

```
        <OBJECTS_GENERATION>
        create_object SHEEP {
            number_of_objects 4
            set_loose_grouping
            set_gaia_object_only
            set_place_for_every_player
            min_distance_to_players 7
            max_distance_to_players 8
        }
```

Listing 11: Give every player 4 gaia sheep close to their starting town.

set_gaia_unconvertible TODO

group_placement_radius NUM Specify the number of tiles out from the central tile that objects belonging to the same group may spawn ($x > 0$, default to 3). Active only when grouping behavior is specified.

set_tight_grouping TODO

set_loose_grouping TODO

terrain_to_place_on TERRAIN The object(s) will only be placed on the specified terrain. The argument passes can be set

```
        <LAND_GENERATION>
        create_land {
            terrain_type DESERT
            number_of_tiles 500
```

```
5           land_position 50 50
6       }
7
8       <OBJECTS_GENERATION>
9           create_object ROCK {
10          number_of_objects 300
11          terrain_to_place_on DESERT
12      }
```

Listing 12: Example where place decorative rocks on a central desert..

layer_to_place_on TERRAIN TODO

place_on_forest_zone TODO

avoid_forest_zone NUM TODO

avoid_cliff_zone NUM TODO

min_distance_to_players NUM and max_distance_to_players NUM Minimum distance (in tiles) from the origin of player lands where that the object can be placed. It is not necessary to specify both attributes. If the objects are grouped, distance refers to the center of the group, not the individual members. When used with place_on_specific_land_id, distances refer to that specific land. When used without set_place_for_every_player or place_on_specific_land_id, maximum distance has no effect.

There is a bug in Age of Empires II: DE such that if distances are very constrained (i.e., $min = max$), objects are noticeably biased towards being placed in the west (i.e., starting villagers). Furthermore, when used with set_place_for_every_player or place_on_specific_land_id, minimum distance applies to **all** lands, not just player lands (or the specific ID).

```
1       <OBJECTS_GENERATION>
2       create_object SCOUT {
3           set_place_for_every_player
4           min_distance_to_players 7
5           max_distance_to_players 9
6       }
```

Listing 13: Place the starting scout at a distance of 7-9 tiles

max_distance_to_other_zones NUM TODO

min_distance_group_placement NUM Minimum distance in tiles that individual objects of the same create_object command, and *all* future objects, must stay away from each object.

Best used with small values, to keep different resources from being directly next to each other. To scatter objects from the same command far away from each other, use temp_min_distance_group_placement. If the objects are grouped, distance applies to whole groups, and refers to the center.

```
1        <OBJECTS_GENERATION>
2        create_object FORAGE {
3            number_of_objects 7
4            number_of_groups 2
5            set_tight_grouping
6            set_place_for_every_player
7            set_gaia_object_only
8            min_distance_to_players 8
9            max_distance_to_players 10
10           min_distance_group_placement 4
11       }
```

Listing 14: Give each player two sets of forages and make them avoid each other by 4 tiles, and keep all future objects 4 tiles away.

temp_min_distance_group_placement NUM TODO

find_closest TODO

force_placement TODO

actor_area ID TODO

actor_area_radius NUM TODO

actor_area_to_place_in ID TODO

avoid_actor_area ID TODO

avoid_all_actor_areas TODO

max_distance_to_players NUM

.16 Functions

The section provides the utility functions you can use within RMS file.

Map size name	Macro to check
Tiny	TINY_MAP
Tiny	SMALL_MAP
Tiny	MEDIUM_MAP
Tiny	LARGE_MAP
Tiny	HUGE_MAP
Tiny	GIGANTIC_MAP
Tiny	LUDIKRIS_MAP

Table 2: Macro to check if you want to detect the map size

.16.1 Random

If you need to generate a random number, use `rnd` function.

1. min: the minimum number that the function can generate;
2. max: the maximum number that the function can generate;

It generates a number $x \in [min, max]$. The example in Listing .16.1 yields numbers in the set $\{5, 6, 7\}$.

```
1         foobar rnd(5,7)
```

.16.2 Choose a randomly chosen scenario

If you need to perform a different command depending on a randomly generated number, you can use the construct start_random, end_random, as shown in Listing .16.2: such a construct will set `FOO` to 3 in the 25% of the cases, 4 in the 25% of the cases and 5 in the remainder 50% of cases.

```
1         start_random
2             percent_chance 25 #const FOO 3
3             percent_chance 25 #const FOO 4
4             percent_change 50 #const FOO 5
5         end_random
```

.17 Inherited constants from the environment

When building a map, it is possible to use conditional in order to check what are the parameters of the map that we need to generate. For example, you may want to obtain how big the map to generate is (e.g., tiny, normal, ludicruous). To do so, you can check whether one of the macro in Table 2 is defined.

.18 Popular included files

.18.1 thebr_setup.inc

Seems empty

.18.2 F_seasons.inc

You can include `F_seasons.inc` file. Such a file changes the texture of the random map, depending on which macros are defined: Table 3 shows which macros should be define in order to tweak the aesthetic of the map. Usually you need to first define one of the macros and then import `F_seasons.inc` (via include_drs).

Macro	Description
PH_ALPINE	
PH_ALPINE_B	No leaves allowed
PH_SPRING	
PH_SPRING_C	
PH_MEDISOUTH	
PH_SPRING_B	No Leaves allowed
PH_TROPHICALSOUTH	
PH_TROPHICALSOUTH_B	Jungle read instead of grass 2
PH_TROPHICALEAST	
PH_DESERT	
PH_AFRICAN	
PH_ASIAN_B	Dry grass on layer C replaced by grass 1
PH_ASIAN	If no season is chosen, this one is used
PH_AUTUMN	
PH_AUTUMN_B	
PH_FROZEN	
PH_AFRICAN_B	cracked instead of Savannah for Layer A, terrain 6 not used
PH_AFRICAN_C	PALM DESERT instead of ACACIA for main forest
PH_AFRICAN_D	swap dirt 4 and 6 + baobab swap
PH_AFRICAN_E	replacing acacia trees with palm desert or dragon trees entirely, leaving only stragglers

Table 3: Available seasons in Age of Empires II: DE.

.19 Constants

In this section there are described all the constants that you may use in a RMS file.

.19.1 Terrain Type

Values used in base_layer command. A full detailed list of how to use terrains is available at `https://docs.google.com/document/d/e/2PACX-1vR_I1I9u-hI2FFm-EYyjoM_-9dNJFOfTaIgr05wXNbdpv9tI18b6r18ARULy3Jqyvlq6-lc2VjX6xP4/pub#h.3bdjnf7tryyk`. Still, some default terrains are shown in Table 4 as well.

ID	RMS Name	Wololo Kingdom Name	DE Scenario Editor	HD Scenario Editor	UP 1.5 Scenario	UP 1.0 Scenario Editor	HD Texture file	DE Texture file	Comment
0	GRASS	GRASS	Grass 1	Grass 1	Grass 1	Grass	g_grs	g_grs	default terrain
1	WATER	WATER, DLC_WATER5	Water, Shallow	Water, Shallow	Water, Shallow	Water, Shallow	g_wtr	g_wtr	dockable
2	BEACH	BEACH, DLC_BEACH2, DLC_BEACH3, DLC_BEACH4	Beach	Beach	Beach	-	g_bch	g_bch	automatically placed when most terrains border water; can build walls on; navigable
3	DIRT3	DIRT3, DIRT4, DLC_DIRT4	Dirt 3	Dirt 3	Dirt 3	Dirt 3	g_ds3	g_ds3	grassy
4	SHALLOW	SHALLOW, DLC_NEW-SHALLOW	Shallows	Shallows	Shallows	Shallows	g_sha	g_sha	walkable and navigable; no buildings
5	LEAVES	LEAVES, DLC_JUN-GLELEAVES	Under-brush	Leaves	Leaves	Leaves	g_for	g_for	used as the underlying texture for many forest types
6	DIRT	DIRT	Dirt 1	Dirt 1	Dirt 1	Dirt 1	g_des	g_des	brown with the occasional cactus
7	-	-	Farm	Farm	Farm	-	g_fm1	g_fm1	terrain only, no food
8	-	-	Farm, Dead	Farm, dead	Dead Farm	-	g_fm2	g_fm2	terrain only, no food
9	GRASS3	GRASS3, MOORLAND	Grass 3	Grass 3	Grass 3	Grass 3	g_gr3	g_gr3	brownish grass
10	FOREST	FOREST, DLC_RAIN-FOREST	Forest, Oak	Forest	Forest	Forest	g_for	g_for	placed on LEAVES
11	DIRT2	DLC_MAN-GROVESHAL-LOW	Dirt 2	Dirt 2	Dirt 2	Dirt 2	g_ds2	g_ds2	dirt/grass mixture

12	GRASS2	GRASS2, DLC_JUNGLEGRASS	Grass 2	Grass 2	Grass 2	Grass 2	g_gr2	g_gr3	very green grass
13	PALM_DESERT	PALM_DESERT	Forest, Palm Desert	Palm Desert	Plam Desert	Palm Desert	g_pal	g_pal	placed on DESERT
14	DESERT	DESERT, SAVANNAH	Desert, Sand	Desert, Sand	Desert	Desert	g_pal	g_pal	sandy and light colored
15	-	-	Water 2D, Shoreless	Water 2D, Shoreless	Water, Shallow (Other)	-	g_wtr	g_wtr	looks like WATER; navigable; no beaches; not dockable
16	ROCK1	BAOBAB, BAOBAB_FOREST, BAOBABS	Grass, Other	Grass, Other 2	Unknown	-	g_grs	g_grs	looks like GRASS; automatically placed under cliffs; the const is only defined in HD and DE
17	JUNGLE	JUNGLE	Forest, Jungle	Jungle	Jungle	Jungle	g_for	g_for	placed on LEAVES
18	BAMBOO	BAMBOO	Forest, Bamboo	Bamboo	Bamboo	Bamboo	g_for	g_for	placed on LEAVES
19	PINE_FOREST	PINE_FOREST	Forest, Pine	Pine Forest	Pine Forest	Pine Forest	g_for	g_for	placed on LEAVES
20	-	DLC_MANGROVEFOREST	Forest, Oak Bush	Oak Forest	Oak Forest	Oak Forest	g_for	g_for	placed on LEAVES; identical to FOREST prior to DE
21	SNOW_FOREST	SNOW_FOREST, DRAGONFOREST	Forest, Pine Snow	Snow Pine Forest	Snow Pine Forest	Snow Pine Forest	g_snf	g_snf	placed on GRASS_SNOW in AoC; placed on a snow/leaves mix in HD and DE
22	DEEP_WATER	DEEP_WATER, DLC_WATER4	Water, Deep	Water, Deep	Water, Deep	Water, Deep	g_wt2	g_wt2	not dockable
23	MED_WATER	MED_WATER	Water, Medium	Water, Medium	Water, Medium	Water, Medium	g_wt3	g_wt3	not dockable
24	ROAD	ROAD	Road	Road	Road	Road	g_rd1	g_rd1	a clean road; cannot place natural resources
25	ROAD2	ROAD2, DLC_DRYROAD	Road, Broken	Road, Broken	Road, Broken	Road, Broken	g_rd2	g_rd2	road broken up by dirt patches; cannot place natural resources

26	-	-	Ice, Navigable	Ice	Ice (Other)	-	g_ice	g_ice	navigable
27	-	DIRT2	Grass, Foundation	Grass, Foundation	Building	-	g_ds2	g_ds2	like DIRT2, no beaches; still dockable; left behind by buildings
28	-	-	Water 2D, Bridge	Water 2D, Bridge	Water, Shallow (Bridge)	-	g_wtr	g_wtr	no beaches; walkable; not navigable; no buildings; produced by bridge objects
29	-	-	Farm, 0%	Farm, 0%	Farm 1	-	g_fc1	g_fc1	terrain only, no food
30	-	-	Farm, 33%	Farm, 33%	Farm 2	-	g_fc2	g_fc2	terrain only, no food
31	-	-	Farm, 67%	Farm, 67%	Farm 3	-	g_fc3	g_fc3	terrain only, no food
32	SNOW	SNOW	Snow	Snow	Snow	Snow	g_sno	g_sno	icy beach when bordering water
33	DIRT_SNOW	ROAD_SNOW, ROAD_SNOWY	-	Snow Dirt	Snow Dirt	Snow Dirt	g_snd	g_snd	icy beach when bordering water
34	GRASS_SNOW	GRASS_SNOW	-	Snow Grass	Snow Grass	Snow Grass	g_sng	g_grs and g_sno	icy beach when bordering water
35	ICE	ICE	Ice	Ice2	Ice	Ice	g_ice	g_ice	not navigable
36	-	DIRT_SNOW	Snow, Foundation	Snow, Foundation	Building (Snow)	-	g_snd	g_snd	like SNOW_DIRT; no beaches; still dockable; left behind by buildings on snowy terrains
37	ICYSHORE	-	Beach, Ice	Ice, Beach	Beach (Ice)	-	g_ice	g_ice	looks like ICE; behaves like BEACH; can place walls; navigable; constant only defined in DE
38	-	CRACKEDIT	-	Road, Snow	Road, Snow	Road, Snow	g_sr1	g_rd2 and g_sno	road with dirt and snow patches; cannot place natural resources
39	-	DLC_JUNGLEROAD	-	Road, Fungus	Road, Fungus	Road, Fungus	g_sr2	g_sr2 and g_des	road with dirt and grass patches; cannot place natural resources
40	DLC_ROCK	DLC_ROCK, QUICKSAND	Rock 1	Rock 1	Road (Other)	-	g_rck	g_rck	no buildings; used for King of the Hill; looks like ROAD in the AoC
41	DLC_SAVANNAH	ACACIA_FOREST	Dirt, Savannah	Savannah	Grass 1 (Other)	-	g_gr5	g_gr5	light brown; buggy unused terrain in AoC
42	DLC_DIRT4	n/a	Dirt 4	Dirt 4	n/a	n/a	g_ds4	g_ds4	dirt with some grass
43	DLC_DRYROAD	n/a	-	Road, Desert	n/a	n/a	g_rd3	g_rd2 and g_pal	road with sand patches; cannot place natural resources

44	DLC_MOOR-LAND	n/a	-	Moorland	n/a	n/a	g_gr4	g_gr4 and g_grs	mud with some grass
45	DLC_CRACKED	n/a	Desert, Cracked	Desert, Cracked	n/a	n/a	g_pal1	g_pal1	buildings take 25% more damage
46	DLC_QUICK-SAND	n/a	Desert, Quicksand	Quicksand	n/a	n/a	g_qs	g_qs	no buildings; no natural resources
47	DLC_BLACK	n/a	Black	Black	n/a	n/a	g_bla	g_bla	completely black; no buildings
48	DLC_DRAGON-FOREST	n/a	Forest, Dragon Tree	Dragon Tree Forest	n/a	n/a	g_des	g_des	placed on DIRT
49	DLC_BAOBAB-FOREST	n/a	Forest, Baobab	Baobab Forest	n/a	n/a	g_ds4	g_ds4	200 wood per tree; 25% tree density; placed on DLC_DIRT4
50	DLC_ACACI-AFOREST	n/a	Forest, Acacia	Acacia Forest	n/a	n/a	g_gr5	g_gr5	150 wood per tree; 50% tree density; placed on DLC_SAVANNAH
51	DLC_BEACH2	n/a	Beach, White, Vegatation	Beach, White, Vegatation	n/a	n/a	g_bc4	g_bc4	behaves like BEACH
52	DLC_BEACH3	n/a	Beach, Vegetation	Beach, Vegetation	n/a	n/a	g_bc2	g_bc2	behaves like BEACH
53	DLC_BEACH4	n/a	Beach, White	Beach, White	n/a	n/a	g_bc3	g_bc3	behaves like BEACH
54	DLC_MAN-GROVESHAL-LOW	n/a	Shallows, Mangrove	Shallows, Mangrove	n/a	n/a	g_sh3	g_sh3	building possible; navigable; not dockable
55	DLC_MAN-GROVEFOREST	n/a	Forest, Mangrove	Mangrove Forest	n/a	n/a	g_sh4	g_sh4	80% tree density; placed on DLC_MANGROVESHALLOW
56	DLC_RAINFOR-EST	n/a	Forest, Rainforest	Rainforest	n/a	n/a	g_fo2	g_fo2	looks similar to JUNGLE; placed on DLC_JUNGLELEAVES
57	DLC_WATER4	n/a	Water, Deep Ocean	Water, Deep Ocean	n/a	n/a	g_wt4	g_wt4	not dockable; darker than DEEP_WATER
58	DLC_WATER5	n/a	Water, Azure	Water, Azure	n/a	n/a	g_wt5	g_wt5	dockable; brighter than WATER
59	DLC_NEWSHAL-LOW	n/a	Shallows, Azure	Shallows, Azure	n/a	n/a	g_sh2	g_sh2	bright blue; behaves like SHALLOW

60	DLC_JUNGLE-GRASS	n/a	Grass, Jungle	Grass, Jungle	n/a	n/a	g_gr6	g_gr6	dark green
61	DLC_JUNGLEROAD	n/a	-	Road, Jungle	n/a	n/a	g_rd4	g_sr2 and g_gr6	road covered in grass; cannot place natural resources
62	DLC_JUNGLE-LEAVES	n/a	-	Leaves, Jungle	n/a	n/a	g_fo2	g_fo2 and g_gr6	mixture of LEAVES and DLC_JUNGLEGRASS
63	-	n/a	Rice Farm	Rice Farm	n/a	n/a	g_rm1	g_rm1	no food; building possible; navigable; not dockable
64	-	n/a	Rice Farm, Dead	Rice Farm, Dead	n/a	n/a	g_rm2	g_rm2	no food; building possible; navigable; not dockable
65	-	n/a	Rice Farm, 0%	Rice Farm, 0%	n/a	n/a	g_rc1	g_rc1	no food; building possible; navigable; not dockable
66	-	n/a	Rice Farm, 33%	Rice Farm, 33%	n/a	n/a	g_rc2	g_rc2	no food; building possible; navigable; not dockable
67	-	n/a	Rice Farm, 66%	Rice Farm, 66%	n/a	n/a	g_rc3	g_rc3	no food; building possible; navigable; not dockable
68	-	n/a	-	-	n/a	n/a	g_r01	g_r01	uses the classic grass texture
69	-	n/a	-	-	n/a	n/a	g_r01	g_kf1	used for battle royale; visible through fog of war; builable; does not cause damage
70	-	n/a	Gravel, Default	Moddable Land 70	n/a	n/a	g_m00	g_gravel_default	grey gravel
71	-	n/a	Underbrush, Leaves	Moddable Land 71	n/a	n/a	g_m01	g_underbrush_leaves	similar to LEAVES
72	-	n/a	Underbrush, Snow	Moddable Land 72	n/a	n/a	g_m02	g_snf	leaves/snow mixture; used by snowy forest terrains
73	-	n/a	Snow, Light	Moddable Land 73	n/a	n/a	g_m03	g_sno	snow that layers weakly with terrain_mask
74	-	n/a	Snow, Strong	Moddable Land 74	n/a	n/a	g_m04	g_sno	snow that layers strongly with terrain_mask

75	-	n/a	Road, Fungus	Moddable Land 75	n/a	n/a	g_m05	g_sr2	very mossy road; cannot place natural resources
76	-	n/a	Dirt, Mud	Moddable Land 76	n/a	n/a	g_m06	g_gr4	brown mud
77	-	n/a	Underbrush, Jungle	Moddable Land 77	n/a	n/a	g_m07	g_fo2	greenish leaves
78	-	n/a	Road, Gravel	Moddable Land 78	n/a	n/a	g_m08	g_rd5	road with gravel; no resource restrictions
79	-	n/a	Beach (Non-Navigable)	Moddable Land 79	n/a	n/a	g_m09	g_bch	looks like BEACH; not navigable; buildable
80	-	n/a	Beach (Non-Navigable), Wet Sand	Moddable Land 80	n/a	n/a	g_m10	g_beach_wet	looks like DLC_WETBEACH; not navigable; buildable
81	-	n/a	Beach (Non-Navigable), Wet Gravel	Moddable Land 81	n/a	n/a	g_m11	g_gravel_wet	looks like DLC_GRAVELBEACH; not navigable; buildable
82	-	n/a	Beach (Non-Navigable), Wet Rock	Moddable Land 82	n/a	n/a	g_m12	g_rock_wet	looks like DLC_WETROCKBEACH; not navigable; buildable
83	-	n/a	Grass, Jungle	Moddable Land 83	n/a	n/a	g_m13	g_gr6	lush green; like DLC_JUNGLEGRASS but with slightly different plant coverage
84	-	n/a	-	Moddable Land 84	n/a	n/a	g_m14	o_mod	
85	-	n/a	-	Moddable Land 85	n/a	n/a	g_m15	o_mod	
86	-	n/a	-	Moddable Land 86	n/a	n/a	g_m16	o_mod	
87	-	n/a	-	Moddable Land 87	n/a	n/a	g_m17	o_mod	

88	MEDITERRANEAN_FOREST	n/a	Forest, Mediterranean	Moddable Land 88	n/a	n/a	g_m18	g_for	mixture of cypress, olive and italian pine trees; placed on LEAVES
89	-	n/a	Forest, Bush	Moddable Land 89	n/a	n/a	g_m19	g_for	bushes; 88% tree density; placed on LEAVES
90	-	n/a	Forest, Reeds (Shallows)	Moddable Beach 90	n/a	n/a	g_m20	g_sha	50 wood; placed on SHALLOW; looks like SHALLOW on the minimap
91	DLC_REEDS-BEACH	n/a	Forest, Reeds (Beach)	Moddable Beach 91	n/a	n/a	g_m21	g_beach_wet	50 wood; placed on DLC_WET-BEACH
92	-	n/a	Forest, Reeds	Moddable Walkable Shallows 92	n/a	n/a	g_m22	g_for	50 wood; placed on LEAVES
93	-	n/a	-	Moddable Walkable Shallows 93	n/a	n/a	g_m23	o_mod	
94	-	n/a	-	Moddable Walkable Shallows 94	n/a	n/a	g_m24	o_mod	
95	-	n/a	Water, Green	Moddable Shallow Water 95	n/a	n/a	g_m25	g_wt_green	dockable
96	-	n/a	Water, Brown	Moddable Shallow Water 96	n/a	n/a	g_m26	g_wt_brown	dockable
97	-	n/a	-	Moddable Water 97	n/a	n/a	g_m27	o_mod	
98	-	n/a	-	Moddable Water 98	n/a	n/a	g_m28	o_mod	
99	-	n/a	-	Moddable Deep Water 99	n/a	n/a	g_m29	o_mod	

100	DLC_DRY-GRASS	n/a	Grass, Dry	n/a	n/a	n/a	n/a	g_gr7	brownish grass
101	DLC_BOGLAND	n/a	Swamp, Bogland	n/a	n/a	n/a	n/a	g_qs2	blueish grass; no buildings; no natural resources
102	DLC_DESERT-GRAVEL	n/a	Gravel, Desert	n/a	n/a	n/a	n/a	g_ds5	lighter version of gravel
103	DLC_ROAD-GRAVEL	n/a	-	n/a	n/a	n/a	n/a	g_rd5 and g_des	gravel road with dirt patches
104	DLC_FORESTAUTUMN	n/a	Forest, Autumn	n/a	n/a	n/a	n/a	g_for	placed on LEAVES
105	DLC_FOREST-SNOWAUTUMN	n/a	Forest, Autumn Snow	n/a	n/a	n/a	n/a	g_snf	snowy version of DLC_FORESTAUTUMN; placed on underbrush snow
106	DLC_FOREST-DEAD	n/a	Forest, Dead	n/a	n/a	n/a	n/a	g_snf	snowy bushes and dead trees; placed on underbrush snow
107	DLC_WET-BEACH	n/a	Beach, Wet	n/a	n/a	n/a	n/a	g_beach_wet	like BEACH but darker
108	DLC_GRAVEL-BEACH	n/a	Beach, Wet Gravel	n/a	n/a	n/a	n/a	g_gravel_wet	behaves like BEACH
109	DLC_WETROCK-BEACH	n/a	Beach, Wet Rock	n/a	n/a	n/a	n/a	g_rock_wet	behaves like BEACH

Table 4: List of all the default terrain types available

.19.2 Map Type

Values used in ai_info_map_type. If the map deviate too much from the standard maps, consider using "CUSTOM" or leaving out the command entirely.

Name	Value	Note
ARABIA	9	
ARCHIPELAGO	10	
ARENA	29	
BALTIC	11	
BLACK_FOREST	12	
COASTAL	13	
CONTINENTAL	14	
CRATER_LAKE	15	
FORTRESS	16	
GHOST_LAKE	32	
GOLD_RUSH	17	
HIGHLAND	18	
ISLANDS	19	
KING_OF_THE_HILL	30	
MEDITERRANEAN	20	
MIGRATION	21	
MONGOLIA	26	
NOMAD	33	
OASIS	31	
RIVERS	22	
SALT_MARSH	28	
SCANDANAVIA	25	
TEAM_ISLANDS	23	
YUCATAN	27	
STEPPE	34	added in HD
BUDAPEST	35	added in HD
VALLEY	36	added in HD
ATLANTIC	37	added in HD
LAND_OF_LAKES	38	added in HD
LAND_NOMAD	39	added in HD
CENOTES	40	added in HD
GOLDEN_HILL	41	added in HD
MEGARANDOM	42	added in HD
MICHI	43	added in HD
AMBUSH	44	added in HD
CUSTOM	45	added in HD
NILE_DELTA	46	added in HD
MOUNTAIN_PASS	47	added in HD
SERENGETI	48	added in HD
SOCOTRA	49	added in HD
KILIMANJARO	50	added in HD

Table 5: List of all the map types available

Terms

Personal Computer A general purpose machine. vii

Bibliography

[1] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[2] [ĐĐ] Krampus. Bug: Stable keeps disappearing with video (now with how to fix it!). https://steamcommunity.com/app/813780/discussions/2/2145343448814946626/?ctp=4#c2260186248407607030. (accessed: 28.12.2020).

[3] Zetnus. Random map scripting links and faq. http://aok.heavengames.com/cgi-bin/forums/display.cgi?action=ct&f=28,42485,,30. (accessed: 30.12.2020).

[4] Ozhara. Ozhara's modding guide for aoe 2 hd. https://steamcommunity.com/sharedfiles/filedetails/?id=190070326. (accessed: 28.12.2020).

[5] Yorok0. A guide to modding? https://forums.ageofempires.com/t/guide-to-modding/61109/4. (accessed: 28.12.2020).

[6] Zetnus. Definitive random map scripting guide. https://-docs.google.com/document/d/e/2PACX-1vR_I1I9u-hI2FFm-EYyjoM_-9dNJFOfTaIgr05wXNbdpv9tI18b6r18ARULy3Jqyvlq6-lc2VjX6xP4/pub. (accessed: 29.12.2020).

[7] TurnupHyperion4. A guide to modding? https://forums.ageofempires.com/t/guide-to-modding/61109/2. (accessed: 28.12.2020).

[8] A Fandom user. Units. https://agecommunity.fandom.com/wiki/Units. (accessed: 28.12.2020).

[9] Estien Nifo Mikko, Apre. Advanced genie editor. Advanced Genie Editor 2020.3.30.

[10] Nelmatd. Graphics. https://agecommunity.fandom.com/wiki/Graphics. (accessed: 28.12.2020).

[11] Zetnus Ryan Andrews, Bultrom. Age of empires rms guide. http://aoe.dragonferocity.com/guide/. (accessed: 29.12.2020).

[12] Forgotten Empires. Rms features. https://www.forgottenempires.net/age-of-empires-ii-definitive-edition/rms-features. (accessed: 30.12.2020).

[13] Ensemble Studios. Scripting guide - userpatch v1.5. http://userpatch.aiscripters.net/reference.html#. (accessed: 30.12.2020).

[14] Zetnus. Rms tutorials: All about cliffs. http://aok.heavengames.com/cgi-bin/forums/display.cgi?action=ct&f=26,42261,,all. (accessed: 30.12.2020).

[15] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.