Chronicle

Manual

This file is accurate as of version 3.2.

Chronicle is a utility designed to read, parse and display information stored in savegame files. Data is presented on maps, in tables and graphs. It can be saved or further exported as graphics or CSV tables. Multiple points of time are supported so that time-based progression of various statistics can be presented. Savegames (autosaves) can be listened to in the background as the game progresses which is an easy way to acquire long time-series for the program to show.

Table of contents

Usage

Game data can be acquired by opening a savegame file of a supported game (see 'Game management'). You can use quick savegame import (through *File → Import*) or navigate to specific savegame file on the disk using *File → Open* method. Further savegame data can be added using *File → Import → Import (add)* button.

After the data is imported, you can either see various savegame statistics displayed on map, in tables and in graphs, in respective tabs.

Games support

The program currently supports the following games:

Game	Ironman	Notes
 **Europa Universalis III**	n/a	Tested on Divine Wind 5.2; will not work on earlier versions of EU3
 **Hearts of Iron III***	n/a	Tested on Their Finest Hour 4.02 (hoi3game.exe)
 **Hearts of Iron III: Their Finest Hour***	n/a	Tested on Their Finest Hour 4.02 (hoi3tfh.exe)
 **Victoria II**	n/a	Tested on Heart of Darkness 3.03
 **Crusader Kings II**	Y	Tested on version 2.5
 **Europa Universalis IV**	Y	Tested on version 1.14; currently not compatible with Random New World from Conquest of Paradise expansion

* Support for Hearts of Iron III is divided into two games – you should use 'Their Finest Hour' if you have this extension installed, run it and/or run any mods that use it. It then uses files from 'tfh' subfolder, rather than from the main game folder.

Interface

'File' menu

- *New* - unloads the currently loaded game data;
- *Open* - opens an AAR file or imports a savegame file, depending on the type of file that is selected; replaces all the data currently loaded and initializes a new game, if necessary;
- *Save* - saves the currently present data into the current AAR file;
- *Save as* - saves the currently present data into a new AAR file;

Import

- *Games* – clicking on a game title loads the game data, allows to start a savegame recording session and displays a list of savegames; there are the following options available:
  - *Add* – adds a new game
  - *Edit* – edits a selected game entry
  - *Remove* – removes a game entry (no actual game files are deleted)

- *Import* – imports all the games from Steam directory.
  Installed games information is used when importing savegames to apply game definitions (countries, provinces data, etc.). Clicking on a game initializes its environment and allows to see map and choose general (savegames-independent) map modes.

- *Record* – allows to start and stop recording sessions (see: Recording games section) ; the program will monitor locations related to the game that is currently selected in the Games tab; recorded savegames are appended to the currently present ones.
- Savegames – allows quick import of savegame data for the currently loaded game:
  - Import – imports a savegame and replaces all the currently loaded data
  - Import (add) – imports a savegame and adds it to the currently present timeline
  - Delete – deletes a savegame file from the disk.
    Double-clicking on a savegame is the same as choosing an Import method.
    On the savegame list  icon denotes a compressed savegame;  icon denotes a binary savegame (used e.g. with Ironman games) and  icon denotes a text-based savegame.

Update

- Update – if there is an update available, it allows the program to download it and start the setup;
- Settings – specifies how the program should check for updates.

Options

Map options:

- Map labels – choose to turn labels off, display province ID, province name, or value for the current map mode; additionally labels shadows can be turned on or off;
- Animated map export: Size reduction – choose if an animated map should be saved to GIF file at full size or reduced; GIF files employ lossless compression so for unreduced maps containing many recorded years may take several megabytes of disk space;
- Animated map export: Attach information bar – an additional bar may be added to exported GIF files, including information about date of the current frame, map mode and application logo (which lets spread the word about this app);
- Colors options – select colors for borders between provinces (between land provinces, between sea provinces and between land and sea provinces – 'shores') or countries; note that partial transparency may be used; border visibility may also be turned on/off here;

Graph options:

- Top entities displayed - specify how many entities (provinces or countries, depending on the chosen table) should be presented on a graph;
- Entities selection – choose how those top entities should be selected; the program can take either top x entities looking at the value at the beginning of the recorded period, or at the end of the period, or can calculate average values;

'mixed' setting mixes all of the above and chooses a few entities from each method;

- Graph export size – choose if graphs should be exported as they look (unchanged size) or should there have predefined size;
- Colors options – select colors for graph lines;

Tables options:

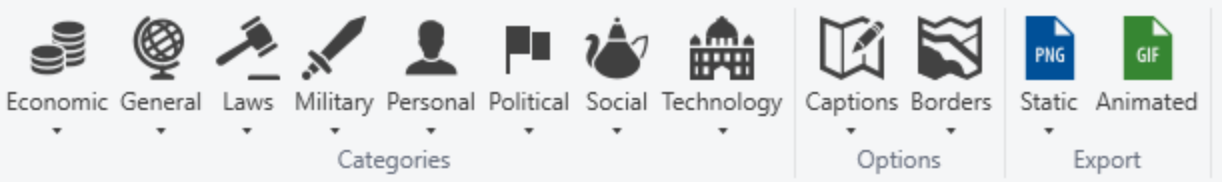
- Compacting – the tables may be set not to display rows that are all empty (have all values of 0), remove rows for countries that do not exist in any point during the recorded play; and not to show water provinces in tables.
- Precision – the precision (number of significant digits in fractions) can be set.

Other options:

- Compression level – set amount of compression applied to AAR files that are saved; dynamic setting means that the bigger the file, the **less** compression is used (files spanning several decades tend to be big and using compression makes handling data considerably slower).

Settings are saved when you navigate away from the settings menu. You may choose 'Revert' button to undo changes made beforehand or click 'Default' to set the settings to their default values.

'Map' tab



- Categories: data modes, which display different data on the map using colors, can be selected here;
- Options: Captions: provides a choice whether to display captions on the map (in the form of province names, province numerical IDs or current mode values) or not to display captions at all;
- Options: Borders: turns on/off province borders shown on the map;
- Export: Static: allows to export the whole world map (in 100% size) or just a currently visible portion (in the current zoom level) to an external image; visual options (borders, shadows, etc.) and current map mode choice are applied in the latter case.
- Export: Animated: exports a current table set to an animated GIF throughout all the timestamps; current map mode is used;
- Refresh: Refresh: trigger a full map refresh (possibly randomizing colors in some map modes);

'Tables' tab

- Categories: data modes can be selected here;
- Type: you can choose between standard and advanced type of tables which are fundamentally different ways to present game data.
  In standard tables, entities (provinces or countries) occupy rows and successive time points are presented in columns to create a two-dimensional table. This allows for quick data analysis in the program interface and enables to create graph out of this data.
  Example:
  Money

Country	1862-08-01	1862-09-01	1862-10-01
China (CHI)	86431.84955	97179.421875	107551.078125
The United Kingdom (ENG)	65027.67569	64829.55859375	82225.546875
Guangxi Clique (GXI)	62502.9213	67404.0078125	72796.6015625
Austria (AUS)	31279.02927	28709.51171875	37813.140625
Yunnan Clique (YNN)	24401.06332	25419.794921875	26111.4375
Prussia (PRU)	22574.98337	29915.41015625	36663.93359375

  The second type, 'advanced' tables, present all the data in rows where successive time points, entities (provinces or countries) and additional data keys are presented one under another. This may be a better choice for export to a spreadsheet program to generate a pivot table there. <u>This type is the only possible choice for multivalue tables.</u>

  Example:
  Money

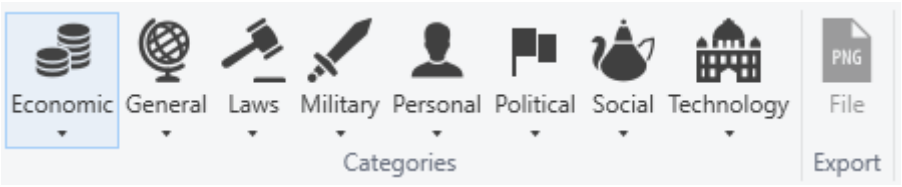
Date	Province	Data key	Value
1862-08-01	The United Kingdom (ENG)		65027.67569
1862-08-01	Russia (RUS)		4876.07227
1862-08-01	France (FRA)		10772.58102
1862-08-01	Prussia (PRU)		22574.98337

  (and further down…)

1862-10-01	Russia (RUS)		9490.232421875
1862-10-01	France (FRA)		3632.095703125
1862-10-01	Prussia (PRU)		36663.93359375

  For what multivalue tables and data keys are, consult 'Multivalue tables' section.
- Export: Clipboard: copies a generated table to the clipboard;
- Export: File: exports a generated table to a CSV file;

'Graphs' tab



- Categories: data modes can be selected here;
- Export: File: exports a graph to an image file.

Game management

When reading savegame file the program uses actual environment definitions located in game folders which do not come in savegames (maps, provinces, countries, etc.). To know where to look for game files, games need to be defined through *File* → *Import* dialogs. If you have different copies of the game folder for the purpose of installing mods (not inside 'mod' folder), you can define them as separate games.

Reading savegames

Before importing a savegame file, the program will initialize the game environment, i.e. map, province and country information and other definitions. Unless the savegame file contains any information about the mods it was played with, the program will determine which game files to read by savegame location (reading a savegame located in a mod folder will prompt the program to look for modded files). In case of lack of conclusion (e.g. reading savegame from beyond the game folders), vanilla files will be used. Various game files are interpreted at this stage: map files (defining provinces placement, terrain, rivers), provinces and countries definitions as well as localization files.

The savegame itself is then parsed according to procedures of the current version of Chronicle. It means that not all of data from the file is acquired but only those figures that fill currently implemented data tables.

AAR file support

The program allows to write imported data into its own format of database (AAR files) which may be saved to disk later read again. The database may contain multiple time points. The file will hold only data imported from savegames into data tables, small part of the whole data contained in savegames, so it will not be possible to recreate a whole savegame at a later point. If newer versions of Chronicle introduce more data tables, these data tables will be filled with blank (default) values when an older AAR file is read.

AAR file contains all the game definitions data so having exactly the same mod environment or having the game installed on the computer at all is not necessary for the file to work.

Recording games

To gather data throughout many game-years, the preferred way is to run the program side-by-side with a game and let it 'listen' to changes of autosave file in background. After the program detects that the autosave file changed (meaning that another period passed in game), the file is automatically parsed and data added to time series. Note that you have to enable autosaving in

game options and data will be added exactly as often as autosaves are made. There is no other method of extracting game data used by Chronicle.

Recording games requires the respective game environment to be initialized (e.g. by clicking on the game in 'Games' tab or opening AAR file or a savegame already related to this game). Recorded time series will be added to already existing ones so you can start recording the game once, save the progress in AAR file and close the game. Later, you can open AAR file and continue recording.

The program listens to autosave changes in all mod folders. This may lead to potential inconsistencies if games from different mods will get recorded within a single AAR file. It is advisable to keep time series from only single mod configuration within one file.

Data presentation

All the data can be presented in tables. Most of tables allow for only single value to be applied to each entity (province or country). E.g. in gold/money data modes each country has a single treasury and therefore only single figure to present.

Multivalue tables

An exception to this are 'multivalue' tables which allow for more than one value to apply to each entity. For example, in Victoria II, each province does not have a single religion but there is more detailed breakdown where each part of the population (so called POP) has its own characteristics. Therefore it is possible that 80% of population follows one religion and 20% another and multivalue tables allow to present this data on the map and in tables.

When a map mode of multivalue table is active, there is additional 'Type' combobox displayed which lets to choose how data should be displayed:

- display top key (uniform): only color corresponding to the dominating feature in each entity (province or country) is shown;
- display top key (shaded): only color corresponding to the dominating feature in each entity is shown but its intensity depends on how much percent the given feature has in given entity ranging from full color (100%) to grey (0%);
- display all keys (shaded): all colors for each features are blended depending on their breakdown in each entity; e.g. if one feature has 80% strength and another 20% strength, their colors will mix but the color of the first feature will be correspondingly dominant;
- display selected key (shaded): another combo box for possible features ('data keys') appears so you can choose interesting feature; all entities show how much of that feature they possess, ranging from full color (100%) to grey (0%).

If 'Selected key' option is active, a key selection box is displayed. Also, there is a color option which means that the shade intensity will depend on the selected feature value in comparison to:

- absolute: the highest value of any key in the world;
- absolute (key): the highest value of selected key in the world;
- relative (province): the sum of all keys in the given entity.

If ‘Use key color’ box is checked, default red key color will be used instead of the selected key own color which may improve readability.

Note that for multivalue tables graphs cannot be drawn in Chronicle. It is best to export the data to a spreadsheet program and analyze them further in pivot tables.

Graphs

Basing on the table data, the program also allows to draw simple graphs for single value tables, with time series on X axis and values for selected entities displayed on Y axis. Choice of entities (provinces or countries) depends on settings in Graphs → Options fly-out.

For any extensive analysis, the preferable way is to export tables to a spreadsheet program and draw graphs there.

Disclaimers

All trademarks are property of their respective owners. Especially, names of games are used for informative purposes only. The program is not related in any way to Paradox Interactive AB or any of its affiliates.

The program makes use of the following libraries:

- DotNetZip http://dotnetzip.codeplex.com/;
- OxyPlot http://oxyplot.codeplex.com/;
- Fluent Ribbon Control Suite http://fluent.codeplex.com/;
- Extended WPF Toolkit (http://wpftoolkit.codeplex.com/).

The program uses icons from the icon package from http://icons8.com/.

The program uses LZW encoder for GIF creation and other parts of NGif encoder (http://www.codeproject.com/Articles/11505/NGif-Animated-GIF-Encoder-for-NET).

The program decodes binary savegame files with routines and token lists partially based on the work of PreXident and other contributors for Java Savegame Replayer utility (https://forum.paradoxplaza.com/forum/index.php?threads/utility-java-save-game-replayer.722493/)

Special thanks go to:

- PreXident for publishing his program code which made decoding content of binary saves much easier;

- Lucifer for his invaluable help with DH Companion which experience eventually helped me to make this program as well.