RandomParadox Documentation

P

April 2023

1 Introduction

A tool to randomly generate scenarios for multiple games.
Currently implemented modules:

- Hearts of iron IV (Scenario and MapTool(Generation, Loading, Cutting, Scaling))
- Europa Universalis IV (MapTool(Generation, Loading, Cutting, Scaling))
- Victoria 3 (Partial Maptool, very experimental)

To avoid error reports that are due to incorrect configuration, please read the documentation carefully.

1.1 How to read this document

Read section 2. Then read section 3 and follow its instructions.
Now, with a basic idea of the usage of this program, you can use its advanced features:

- Changing parameters of the RandomParadox generator config, explained in 4
- Changing parameters for the game modules you wish to generate for:
  - Changing parameters of the Hearts of Iron IVModule, explained in 5.1
  - Changing parameters of the Europa Universalis IVModule, explained in 5.3
- Changing parameters of the FastWorldGenerator config, explained in 6. Note that without understanding the features explained in section 6.9, do NOT modify any of the layers (the part in between the brackets after "layers :" in FastWorldGenerator.json).
- Understanding interactive mode, mentioned in section 6.8
- Tuning advanced map generation parameters, explained in section 6.9

2 Modifying the config files

The generator works with config files in the json format, which is a file type with a very strict syntax.
In general, no changes to the fields on the left should be made. Such changes will stop the program before generation.
In general, when working with the config files, a few rules should be obeyed:

- **Never add or remove quotes**
- **Never add or remove opening and closing brackets “{“ or “}“**
- **Never add or remove commas**
- **Never add or remove colons “:“**

There are multiple types of fields:

- **Boolean** fields: They can either be *“true“* or *“false“*, which can be understood as *yes* and *no* or *on* and *off*. Never type anything other than *true* or *false* there.
- **String** fields: *Strings* are surrounded by quotes. Never remove the quotes. Only change what is inside the quotes on the right.
  Example: **“heightmapPath“: “inputs/world_1024_paint.bmp“** can be changed to **“heightmapPath“: “inputs/middle_earth_1024_paint.bmp“**
- **Double** fields: These are numbers in the format e.g. *“2.5“*. More details to the sensible ranges can be found in the documentation for each config file.
- **Integer** fields: These are whole numbers in the format e.g. *“2“*. More details to the sensible ranges can be found in the documentation for each config file.

3 Getting started with a simple setup

This simple example shows how to generate a map mod for Hearts of iron IV. It works very similar in Europa Universalis IV as well.

- After downloading and unpacking the generator, you have a folder containing the executable, a config folder, a MetaConf.json, an inputs folder and a resources folder.
- Open the config file *MetaConf.json* and edit the field *username* to your username.
- Then open the config file *configs/RandomParadox.json* and edit the fields *gamePath* (points to your installations of the games), *modPath*(points to the place you want the mod installed at) and *modsDirectory*(points to the

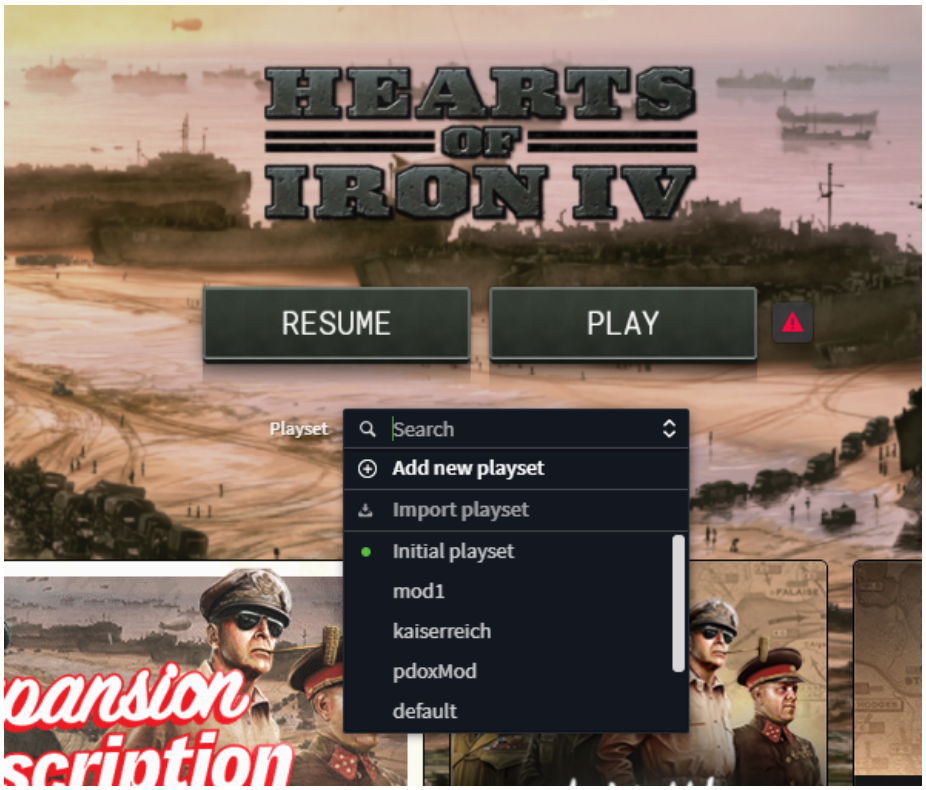

Figure 1: Add a playset to the launcher.

Hearts of Irov IV mod directory, usually under "*C://Users//**YourUsername**//Documents//Paradox Interactive//Hearts of Iron IV//mod*"). If the paths are incorrect, the generator will complain to you.

- Also activate generation by setting the value for *genhoi4* to *true*
- Now you can run the executable. The generation will take a few minutes on default settings, so be patient. The generation is done when the window shows "Done with the generation".
- Now start the game via launcher. In the launcher, create a new playset (see figure 1), add the mod to it (see figure 2), select this playset and launch the game. You should then be able to start a new game. Enjoy!

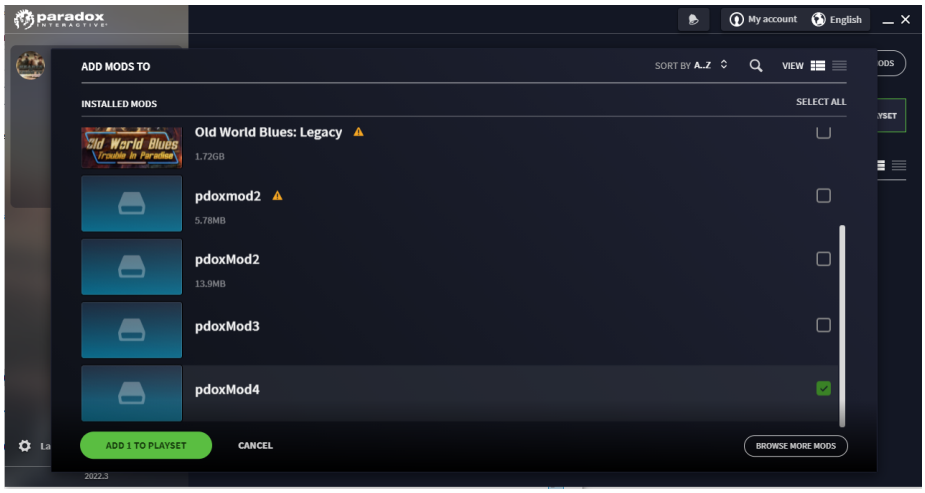

Figure 2: Add mod to the newly created playset.

4 Documentation for RandomParadox Generator

Generic settings are set in the file *RandomParadox.json*.
The list of options:

- **"writeMaps": true**, set to true, if you want the FastWorldGenerator maps created in the "Maps" folder. These maps are not used by the games, but show a lot of info about the world
- **"editmode": false,** set to true if you want to use the map editor. Partially works for Hearts of Iron IV. You can try to launch it, but it will not yet save data and has bugs.
- **"genhoi4": true,** set to true if you want to generate a scenario for Hearts of Iron IV
- **"geneu4": true,** set to true if you want to generate map files for eu4. Not yet feature complete.
- **"genvic3": true,** set to true if you want to generate map files for Victoria 3. Not yet feature complete.
- Every game has settings for paths
  - **modName** : "pdoxMod2". Any name you want. Will create a new folder for every mod name, in case you don't want to overwrite. If the name stays the same, files will get overwritten.

Figure 3: Indonesia cut out from the base Hearts of Iron files and scaled to 5632 * 2048

- **gamePath** : the path to you game directory. The program searches under normal paths, but might not find the game. Please configure this correctly
- **modPath** : the path you want the mod installed at. Doesn't have to be the standard game mod directory. Can be anywhere on your computer.
- **modsDirectory** : MUST point to the standard Game mod directory. If configured incorrectly, the launcher will not show this mod as available. Usually under **Documents//Paradox Interactive//Game Name//mod//**

4.1 Cutting and scaling

If you want to create scenarios on the map of the base game, you can specify so in the cutting options. An example config for cutting out indonesia and scaling it to a normal map size is given in the *configs/cutFromBase* directory. Result can be seen in figure 3

First, cut must be set to true in the *FastWorldGenerator.json*. Furthermore, settings in the FastWorldGenerator settings must be changed. See section 6.7 for more details. Please note: **Only** the *heightmap.bmp* and *provinces.bmp* can be activated via loadMaps. Loading the others will not work.

5 Documentation for the Game Modules

Every game has its game specific file, located under *GameName*Module.json, such as *Hearts of Iron IVModule.json*, FastWorldGenerator overwrites:

- **loadMapsPath**: The path that maps should be loaded from for input to map generation and cutting, if cutting is enabled
- **heightMapName**: the file name of the heightmap you want to generate from. This overwrite makes it possible to load from specific directories for each game.

5.1 Documentation for the Hearts of Iron IV Module

Specific settings for the Hearts of Iron IV Scenario can be found in *Hearts of Iron IVModule.json*. The rules for handling json files apply, see chapter 2.
Hoi4 Settings:

- **strategicRegionSize**: modifier for the size of strategic regions.
- **resourceFactor**: double field. Sensible values between 0.0 and 1000.0. Modifies the total amount of resources available in the game. The higher, the more resources.
- **aluminiumFactor, chromiumFactor, oilFactor, rubberFactor, steelFactor, tungstenFactor**: like resourceFactor, but only for the specific resources. These modifiers work **on top** of resourceFactor. So resourceFactor: 2.0 and oilFactor 2.0 would give 4.0 times the amount of oil. Sensible values between 0.0 and 1000.0.

Scenario settings:

- **numCountries**: how many countries will be placed. An exact result is not guaranteed. Some countries will be generated but not be placed
- **worldPopulationFactor**: multiplier for the world population. Note that this modifies total available manpower. Too high numbers will crash the game, too low numbers will not allow units to be trained. Sensible values probably between 0.1 and 10.0.
- **industryFactor**: multiplier for available industry at game start. Note that this is limited by build slots in the states. From a certain point on, increasing the factor will not have an effect.

5.2 Documentation for the Europa Universalis IV Module

Specific settings for the Europa Universalis IV Scenario can be found in *Europa Universalis IVModule.json*. The rules for handling json files apply, see chapter 2.

Scenario settings:

- **numCountries**: how many countries will be placed. An exact result is not guaranteed. Some countries will be generated but not be placed

5.3 Documentation for the Victoria 3 Module

Specific settings for the Victoria 3 Scenario can be found in *Victoria3Module.json*. The rules for handling json files apply, see chapter 2.

Scenario settings:

- **numCountries**: how many countries will be placed. An exact result is not guaranteed. Some countries will be generated but not be placed

6 Documentation for FastWorldGenerator

Specific settings for the World Generation can be found in *FastWorldGenerator.json*. The rules for handling json files apply, see chapter 2.
Many settings also apply for input heightmaps.

6.1 Module Settings

- **debugLevel**: between 0 and 9. 0 means almost no output, 9 means a lot of debug output
- **loadMaps**: if true, settings in the loadMaps section are used to define whether a map should be loaded instead of generated.
- **writeMaps**: When generating for Paradox Games, leave as true
- **seed**: can be any number from 0 to 2147483647. This determines how the map is generated. If 0, a random seed is chosen. For any other number, the result of the generation will always be the same. Meaning the result of seed 1 will always be the same, but different from the seed 2.

6.2 Loadmap Settings

The loadMaps part includes:

- **heightMap**: defines that a heightmap.bmp must be loaded from the Maps folder
- **terrainMap**: loads terrain.bmp from maps folder. Is **ignored** when heightMap is loaded, because necessary calculations for sealevel and landmass **must** be run
- **humidityMap**: loads humidty.bmp from maps folder
- **habitabilityMap**: loads habitability.bmp from *maps/world/* folder

- **provinceMap**: loads provinces.bmp from maps folder
- **modifyHeight**: false, if set to true this will allow a loaded heightmap to be modified so it fits BOTH to sealevel and landPercentage. This can be useful if a given heightmap has a too low or too high seaLevel, so it will be adjusted via the desired landMassPercentage.
- **heightAdjustments**: Shift the loaded heightmap to lower or higher terrain. This can be used on a loaded map and allows e.g. to generate a flooded earth in combination with **modifyHeight** set to true.

For an example see 6.6

6.3 Edit Settings

Edit mode is a replacement for "interactive mode" of previous versions. There, you can configure an image editor installed on your computer, which is launched on the maps you want to edit during runtime. You can then either edit the map in the editor itself, or, while the editor is open, replace the complete file in the background, if you have it prepared already. For detailed explanation, check sections 6.7.1 and 6.7.2

6.4 Cutting and Scaling Settings

For detailed explanation, check sections 6.7.1 and 6.7.2

6.5 Map Settings

The map generation has generic settings, and then settings for every layer. The generation works with merging multiple layers of fractal noise and creating a heightmap from it. There can be as many layers as you have resources. Only generic settings will be presented in this chapter, advanced use can be found in subsection 6.9.

- **height**: the height in pixels of the generated maps. Note that large maps require a lot of memory.
- **width**: the width in pixels of the generated maps. Note that large maps require a lot of memory.
- **landPercentage**: Set how much land you desire in percent of the world. Range between 0 and 100
- **seaLevel**: Set at which grayscale value the sealevel is. Range between 0 and 255. Note that Paradox Games have a fixed sealevel, so best leave as is.
- **accuracy**: The lower, the faster the program, but the less accurate the results. Range 1 to 100. If you have time, just leave at 100.

- **rivers** :
  - **numRivers**: The amount of desired rivers. Must be greater than or equal to 0
- **trees** :
  - **forestDensity**: The density of trees in a forest terrain type. Range 0.0 to 1.0
  - **jungleDensity**: The density of trees in a jungle terrain type. Range 0.0 to 1.0
  - **savannahDensity**: The density of trees in a savannah terrain type. Range 0.0 to 1.0
  - **grasslandDensity**: The density of trees in a grassland terrain type. Range 0.0 to 1.0
  - **tundraDensity**: The density of trees in a tundra terrain type. Range 0.0 to 1.0
- **mountains** :
  - **hillFactor**: At which point, between seaLevel and highest peak, hills are considered as a terrain type. Range between 0.0 and mountainFactor
  - **mountainFactor**: At which point, between seaLevel and highest peak, mountains are considered as a terrain type. Range between hillFactor and peakFactor
  - **peakFactor**: At which point, between seaLevel and highest peak, peaks are considered as a terrain type. Range between mountainFactor and 1.0

6.5.1 Heightmap settings

- **complexHeight** : true, if you set this to *false*, only the first layer will be used. Recommended to leave at true.
- **layerAmount**: 9, only change this if you've read and understood section 6.9
- **overallFrequencyModifier** : 1.0, the higher this value, the smaller the continents and islands will be. Sensible range from 0.1 to 10.0. Anything smaller or larger will likely significantly reduce stability and success of the generation.
- **removeChunkThreshold** : 8, number of pixels of a land- or seabody, below they are too small to stay on the map. If set to 8, any island of 8 pixels or smaller will be turned into ocean, and any water mass will be turned into lake. This avoid tiny, barely visible islands/lakes.
- detailed explanation of layer settings can be found in subsection 6.9

6.5.2 Provinces Settings

- **auto**: true, if this is true, the values of landProvinceFactor and seaProvinceFactor are ignored and the amount will be automatically calculated based on the size of the map and the amount of ocean and land
- **landProvinceFactor**: Must be greater than 0. Sensible range from 0.1 to 10.0: The higher, the more land provinces. Too high values will be ignored, as the maximum amount of land provinces is 65535 provinces.
- **seaProvinceFactor**: Must be greater than 0. Sensible range from 0.1 to 10.0: The higher, the more land provinces. Too high values will be ignored, as the maximum amount of sea provinces is 65535 provinces.
- **minProvPerRegion**: 3, sets the minimum amount of provinces to be in a region.
- **minProvSize**: 10, the minimum amount of pixels that a province has to consist of.
- **provinceDensityByHabitability** : 100. How much you want province size to be influenced by habitability. 100 means maximum influence of habitability, 0 means none, which means province size is roughly the same in every region of the world.
- **maxProvAmount** : 20000. The maximum amount of provinces that will be generated. Some games don't like too many provinces and simply crash pr produce weird errors

6.5.3 Humidity settings

- **latitudeLow**: what the lower range of the latitude for the generated world should be. Influences climate of the generated world. Lowest value is 0.0, which is equal to 90 degrees south
- **latitudeHigh**: what the upper range of the latitude for the generated world should be. Influences climate of the generated world. Highest value is 2.0, which is equal to 90 degrees north
- **riverHumidityFactor**: how much rivers influence the humidity around them. The higher, the more humid land around rivers will be
- **riverEffectRangeFactor**: how wide the area of influence of rivers is. The higher, the further out humidity is affected by rivers
- **baseHumidity**: general adjustment of humidity. Lower it for a drier planet, increase it for more humidity

6.5.4 Weather Settings

- **baseTemperature**: what the lowest temperature in the generated world will be
- **deviationFactor**: how much temperature deviates depending on continentality. The more continental a province is, the higher the daily deviation. This means that temperatures between night and day differ a lot.
- **temperatureRange**: How warm provinces can be. Added to baseTemperature, depending on location. Meaning max temperature on the generated map is baseTemperature + temperatureRange.

6.6 Loading Mode

If you want to generate the files from a custom heightmap, you can use the config templates in *configs/genGromInput*. There, you must set the path to the directory and filename to generate from. If you want to generate from a map called *middleEarth.bmp*, which is in the directory *inputs*, you must set in the respective *GameName*Module.json, such as Europa Universalis IVModule.json,

- "loadMapsPath": "inputs//"
- "heightMapName": "middleEarth.bmp"

You can also load multiple maps. This allows to customize generation step by step. A quick scenario is explained here:

- Step 1: You run the generation without loading any maps.
- Step 2: You set loadMaps to true, and start editing e.g. the humidity.bmp. Here, you make most of the land more humid, by painting over the dark grey with white.
- Step 3: You run again with loadMaps.humidityMap set to true. Still, you're not happy, because large parts of the world are still far too cold to be inhabitable, and you want cold areas to also be very habitable because e.g. your desired scenario has ice people or whatnot.
- Step 4: You therefore set loadMaps.habitabilityMap to true. You edit the habitabilityMap and paint cold areas white. Now, you run again.
- Step 5: While province density is now okay, you still want to reshape some provinces. Therefore you also set loadMaps.provinceMap to true. You edit the provinces.bmp.
- Step 6: You run again, and now every scenario that is created is based on your custom heightmap, humidity, habitability and provinces.

6.7 Cutting and scaling modes

This section describes how to cut and scale parts of input images. For that, an input folder must be specified in the *FastWorldGenerator.json*. "loadMapsPath" must be set to something like "mapInputs//". Then, in the folder of the executable, you must create a folder named "mapInputs//".

6.7.1 Cutting

There is a template in the cutFromBase folder in config. You can use this as a basis. In the "cut" section:

- **cut** must be set to true to enable the feature
- **minX**, **maxX** must be set to the corresponding pixels in the base image. Open for example the heightmap.bmp in "mapInputs//" with paint or something similar, and look for the x positions that mark the limits to the left and right of the area you want to cut.
- **minY** and **maxY** work the same way, with one exception: They are inverted. If the base image has a height of 2048: a displayed height of 2000 would be 48, and a displayed height of 0 would be 2048.
- the folder under "loadMapsPath" must be filled with the base game heightmap.bmp and provinces.bmp, you can copy them over from anywhere. In the example given above, the directory would be "mapInputs", now containing a provinces.bmp and a heightmap.bmp. Loadmaps must be enabled via *"loadMaps" : true* and the relevant map fields such as *"heightMap" : true* as well.

6.7.2 Scaling

Now, this only cuts from the base images. If you also want to upscale it, set "scale" to true, and enter the size you want, for example scaleX 5632 and scaleY 2048 for the default Hearts of iron IV map size.
Note that by default, the ratio of x to y will not be kept during scaling. So if you scale a 1:1 image to a 2.5 to 1 image, everything will be stretched to a larger width. Setting "keepRatio" to true will only fill the rest with black bars and is not recommended.
So, when scaling to a 2.5:1 image, you should cut out a 2.5:1 area.

6.8 Edit Mode

Edit mode allows to overwrite certain maps during generation. This works by pausing generation at certain steps, and prompting the user to change a file. The user can either change the file by using the image editing software that is started by the tool, or ignore the tool and replace the whole file. When the

user is done changing the file, it is reloaded and used for further generation. Currently mspaint.exe and gimp are supported to be launched by the generator itself.

6.8.1 Example

Currently supported maps for interactiveMode:

- heightMap
- oceanHeatMap
- riverMap
- humidityMap
- habitabilityMap

By setting **interactiveMode** to true and **habitabilityMap** to true, the program stops during generation and shows a message, see figure 4.

Figure 4: Started editor for selected map. Here, we painted a high humidity over the dry sahara

Now the habitability map can be edited, see figure 5 and **??**.

Figure 5: The result of the generation with a green Sahara

6.9 FastWorldGenerator Advanced Use

This subsection explains more details of heightmap, terrain and climate generation.

6.9.1 Individual layer settings

Heightmaps are generated from multiple combined layers. Each layer has different properties:

- **type:** the fractal type that is used
- **landLayer:** if this layer will be applied only to land after the basic continent shape has already been determined
- **fractalFrequency:** the higher, the smaller the fractals
- **fractalOctaves:** affects ruggedness
- **fractalGain:** how rugged the generation is
- **seed:** individual seed for a layer, if 0, it is automatically determined based on the base seed

- **widthEdge:** from when on terrain height falls towards the east/west edges of the map
- **heightEdge:** from when on terrain height falls towards the north/south edges of the map
- **weight:** how much this single layer is weighted relative to the other layers
- **maxHeight:** what the total maximum height present in this layer can be. Should be between 1 and 255
- **minHeight:** what the total maximum height present in this layer can be. Should be between 1 and 255, and smaller maxHeight
- **tanFactor:** how quickly land height increases. Can produce really rugged layers, specifically good for e.g. mountain ranges. Range between 0.00 and 0.26

Individual layers can be inspected in the layers subdirectory.

6.9.2 Editing image files

Quoting `https://hoi4.paradoxwikis.com/Map_modding#Notes`: "Due to not preserving the order of colours in the palette, Paint.net or Microsoft Paint will not work for terrain.bmp or any .bmp files other than provinces.bmp or world_normal.bmp. GIMP or Photoshop is recommended. In case GIMP is used, in the export settings while exporting to BMP on each map, you need to check the 'Disable writing colorspace information' box."

6.9.3 Supplying a heightmap

You can supply a heightmap. It must be a grayscale bitmap image, either in 8 bit or 24 bit format. An example 8 bit file is distributed with the generator files, in "inputs". You can modify this with Gimp or Photoshop. See 6.9.2 for details.

7 Modifying Resource Files

Warning: This chapter in general is addressed at users that are familiar with the functionality of the generator and want to help expanding it. You should never randomly modify files here.

7.1 Flag Generation

Experienced Users: Only experienced users or modders should modify files here. Stability of the generated scenario can be affected, and crashes of the generator can be caused.
Flag generation works as follows:

7.1.1 Flag templates

First, a flag preset is selected from the "resources/flags/flag_presets" folder. Together with the preset, the sidecar .txt is loaded. Such a file contains lines like:

Listing 1: Flag Template Sidecar File

```
#colourGroups;symbolColourGroups;symbolPlacement;symbolX;symbolY;symbolSize;
p1,p2;s1;true;0.025;0.35;0.25;
```

This defines the colour groups used for this template, in this case only two colours are used, one randomly chosen from group p1, one from group p2. Next, the colourGroup for the symbol is chosen. Next, the symbol placement is set to true or false. True allows a symbol, false does not allow a symbol for this flag type. Next, the x and y offsets for the symbol and the size of the symbol are defined. Note that the offset points to the lower left corner of the symbol **template**. Symbol templates often have larger dimensions than initially appears.

7.1.2 Symbol templates

If the flag preset allows a symbol, a symbol template is selected from the "resources/flags/symbol_presets" folder. Together with the preset, the sidecar .txt is loaded. Such a file contains lines like:

Listing 2: Symbol Template Sidecar File

```
#replaceColour
true;
```

This first option simply defines if the colour should be replaced. For example, for complex insignia, changing colours would give ugly results. In this case, setting false should be preferred.

7.1.3 Adding and editing flag and symbol templates

Any template can be edited, and new templates can be added. New templates must be tga-files in the 82x52 dimension for flags and 52x52 dimension for symbol templates. The filename must be a number, that follows the other templates in the folder. Don't forget to add the corresponding side-car text files.

7.2 Name Generation

Experienced Users: Only experienced users or modders should modify files here. Stability of the generated scenario can be affected, and crashes of the generator can be caused. When modifying name generation, you have different options:

7.2.1 Add new state names in state_types.txt

It is important to format them correctly. Separate them via ; and also ALWAYS contain template where the country name is supposed to be.
Example: **Revolutionary template**; However, you cannot add new ideologies/lines like democratic or monarchy. So no new lines.

7.2.2 Editing token groups in token_groups.txt

You can add token groups by adding a new line, e.g. :
mytokenGroup;n;tt;r;s;reallyLongEntry;
First, the name of the group. All entries afterwards are characters that appear randomly if you use this group in name_rules.txt. Each entry can be as long as you want.
You can of course also edit all other token groups, remove and add characters, or remove a group altogether. However, if you remove a group, you need to remove it from name_rules.txt
If you ever see an error mentioning a missing namegroup or token group, make sure to check for errors in the text files. There is no need to for capitalization of letters.

7.2.3 Editing name rules

You can modify or add name rules. Name rules are randomly chosen for countries/regions etc. You can add as many name rules as you wish. Name rules work like this: For every entry separated by ; in the name rule, a random entry from a token group is selected. E.g.: E.g., the name rule: vowels;groupMiddle;vowels;groupEnd; gives:

- a random element from vowels, e.g. A.
- a random element from groupMiddle, e.g. lt.
- a random element from vowels again, e.g. e.
- a random element from groupEnd, e.g. stan.

The full name now is Altestan.

7.3 Hoi4 Unit Generation

7.4 Hoi4 Focus Tree Generation

Only for experienced modders: Working on the focus tree generation is only possible if you understand focus modding in general.
Focus tree generation is done dynamically. For this, resource files from *resources/hoi4/ai/national_focus* are loaded. The folder structure is as follows:

- baseFiles contain the key-value pairs for the **available, bypass and completionRewards** fields in a focus. foci.txt contains information on the foci to be used, and their implemented types in the generation. focusBase is the base template file for every focus tree
- the chains folder contains the chains that are evaluated for every country.
- focusTypes contains the foci templates, defined in foci.txt

7.4.1 Flow of focus generation

Focus generation works with chains. A chain consists of multiple steps, that get evaluated. Each chain step consists of multiple fields:

- chainID: a unique ID for this chain.
- chainStep: Unique ID for the step, used as reference by other steps
- stepRequirements: multiple fields, which are dynamically evaluated.
  - predecessor: names the preceding foci. If the preceding focus could not be filled, this focus won't be created.
  - skippable: Names the preceding foci IDs that can be skipped. Skippable foci are therefore optional in a focus tree.
- rank: Determines which kind of country is allowed to have this focus. Options are **weak, regional, major, any**
- ideology: Determines which ideology the source country must have. Options are **fascism, communism, democratic, neutrality, any**
- type: Specifies the type of the focus, must be one defined in foci.txt. This specifies the template file being used from the focusTypes directory
- target: The requirements the target must fulfill:
  - rank: Determines the rank the destination country must have: Options are **weak, regional, major, any**
  - ideology: Determines the ideology the destination country must have, **relative to the source country** *or* **the specific ideology**: Options are: **not, same, any, fascism, communism, democratic, neutrality**
  - location: How far away the destination target must be. Options are **neighbour, near, far, any**.
  - target: If this country can be selected multiple times on the same level or chain. If for example stepID 0 gives an annex wargoal against a country, step 7 shouldn't give it against the same country again. Options are **notlevel, level, notchain, chain**

- relation: Fields that specify the relations between foci.
  - Exclusive: specify stepID with which this focus is mutually exclusive
  - and: specify stepID which is also required to be fulfilled. If e.g. focus 5 has *predecessor{3,4}*, and 3 has *and{4}*, 4 has *and{3}*, both 3 and 4 must be fulfilled for focus 5 to be available.
  - if nothing is specified in both fields, multiple predecessors are considered as an **OR** relation, meaning any of the preceding foci is enough to proceeed in the focus tree
- Date: The date from which on the focus is available: Format: DD-MM-YYYY, e.g. 01-01-1937.
- available, bypass and completion rewards: Defined here are **keywords**, which specifiy **values** in the *baseFiles/available.txt*, *baseFiles/bypass.txt* and *baseFiles/completion_reward.txt*. These values are explained in the next section.
- level: the last field specifies on which level the focus is. This means that e.g. if you want a focus that has an exclusive relation, e.g. ally fascist state A or annex fascist state A, both are on the same level, so that *targetlevel* can specify that the target MUST be on the same level, so you can guarantee that the SAME country is targeted twice.

7.4.2 Available, bypass and completion_reward files

These files contain key, value pairs of hoi4 national focus code. They key{} field is used in the chain files, to look up the longer value{} fields. An example is given here:

Listing 3: Flag Template Sidecar File

```
key{coastalState}
value{any_state = {
        is_coastal = yes
        is_controlled_by = ROOT
        }};
```

In this example, if a chainstep specifies *available{coastalState}*, this code is placed in the focus. The content here is Hoi4 code, specifying that the country must have at least one coastal state.
This functionality works the same with bypass and completion rewards. Specify the keys in the files together with values, and you're done.
Additional note: Also usable are template keywords: If you for example specify templateSourceTag or templateDestTag, these will be replaced by the source or destination tags during the final creation steps of the focus tree generation.

7.4.3 Issues and lacking functionality

So far, no relations to other chains are possible, only internally in a chain. So you can't specify that a chain is exclusive with another one. In that case, you'd have to build a larger chain incorporating the smaller chains, to work with the step relations.
Furthermore, you can't specify that a focus E has as predecessors (A and B) OR (C and E).
More complex relations might also not work. If you find a relation that doesn't work, please forward it to me.

8 Common Issues

8.1 Common errors

- "Configured paths seem to be messed up, check Hearts of Iron IVModule.json": You have misconfigured paths. Make sure they are in a format like *D://Steam// steamapps//common//Hearts of Iron IV*. The double forward slashes "//" are **important**
- Incorrect config "*.json": The mentioned .json file has an error. The line the error is at should also be mentioned in the message your program shows. Make sure you follow the rules in section 2. If it doesn't work, search for "json validator" and copy the whole file content there.
- "No input map found under heightmapPath, please correct the path": Find and correct the field *heightmapPath* in RandomParadox.json.
- in case the generator can't load one of the maps you edited, make sure you edited them correctly, with the correct tools. See 6.9.2 for details.

9 Known Issues

9.1 Necessary Improvements for FastWorldGen

9.2 Necessary Improvements for Hearts of Iron IVModule

- Focus Tree Generation is lacking features and proper focus trees
- Flag Generation needs more symbol templates, colour combinations and flag templates