Random Map Scripting for AoE II HD

William "HousedHorse" Findlay

December 22, 2018

Contents

Listings

1 About the Author

William “HousedHorse” Findlay has been playing Age of Empires II since his father introduced it to him when William was three years old.

He has played Starcraft II at a semi-professional level, reaching rank number 5 on the North American server. He is also 2100 rated on Age of Empires II HD.

William is an admin of the Age of Empires Memes Community group on Facebook, an Age fan page with over 30,000 members.

He works as a teaching assistant in the Computer Science department at Carleton University in Ottawa, Ontario, Canada. William enjoys scripting and coding and sees random map scripting as a great way to blend his passion for programming and his love of Age of Empires II.

2 What is RMS?

Welcome to my **random map scripting** guide! If you are here, you probably have some general idea of what an RMS is. If not, don’t worry! I will be explaining all about random map scripting in this section and all the great things you can do with it.

When you think about custom games in Age of Empires II, typically a scenario is the first thing that comes to mind. Scenarios are great. You can do lots of cool things in them. The only issue with a scenario is that it is the same thing every time. And if a mapmaker does not take great care, a scenario can easily become too unbalanced for competitive play.

Age of Empires takes care of this problem with the concept of random maps. While a lot of games use “scenarios” for their competitive matches, Age of Empires competitive matches are played **exclusively** on random maps. By learning how to create random map scripts of your own, you can create custom map types for players to enjoy competitively.

Some examples of random maps include Arabia, Arena, Black Forest, Nomad, and many, many more. These are all examples of maps that come with the base game. However, if we journey to the `Custom` section in the lobby screen, we can see a list of **custom random maps**. In this guide, we are going to learn to create maps like these.

2.1 Why is it Called a Random Map Script?

Age of Empires II uses a random map generator to create its random maps. A random map generator is an **algorithm** which takes a random value called a seed and carries out a series of steps to output a full randomized map.

By default, the generator will spit out a flat grassy map with no resources and a town center, scout, and villagers for every player. But this is super boring. How do we make it do interesting things?

Well, it turns out we can write a text file called a **random map script**. This text file is structured in a special way so the random map generator can read it and use it as a guideline for map generation.

For example, we could set the terrain to be snowy instead of the default grass and add randomly placed lakes and resources. The generator will read these instructions and adjust its random output so the map fits our guidelines. Now we’re starting to have something that looks more like a real map!

3 How to Use This Guide

In this section we will detail exactly how to read this guide. We will be going over some simple conventions we use to make reading the guide simpler and following that up with some basic RMS[1] syntax[2] to get you started and thinking about what a proper script looks like.

3.1 Guide Conventions

In this guide, we will be showing script examples throughout. The examples will be shown in the form of code blocks which look like Listing 1.

```
1 /* hello! */
2 /* this is an example code block! */
3
4 /* my custom terrain generation goes here! */
5 <LAND_GENERATION>
6 base_terrain GRASS /* this will create a grassy map! */
```

Listing 1: An example RMS code block.

Each block will be labeled with a **listing number**. This is the number that will be referenced in text. If you see a listing number in the text, you can take a look nearby for the accompanying sample code.

Code blocks like in Listing 1 will be surrounded by a black box. To the left of the blocks, you will see line numbers for your reference. If you are copying the sample code, **do not copy the line numbers**. They are just for your reference.

3.2 Basic RMS Syntax

In this subsection, we will learn about some basic syntax to get you started thinking about what an RMS should look like. Syntax is the way our script has to be formatted in order to allow the game to read it properly. Syntax is **super important** or your script will **break horribly** and you will be sad.

3.2.1 Comments

Comments are bits of text that you can read and understand but the random map generator ignores completely. It is important to comment your script so that you can come back and understand it later.

Definition 3.1. A comment is a block of text that humans can read, but the generator will ignore.

I also like to use comments in my script template so that when I am making a new script I have easy access to all the constants and functions I need. I will be going over my template much later in the guide.

So, how do we write a comment? It's pretty easy. Comments start with `/*` and end with `*/` . The `/*` and `*/` do not need to be on the same line and in this way you can write a comment that spans multiple lines. A comment can be as short or as long as you like. See Listing 2 for an example.

As you can see in Listing 2, it's pretty easy to make a comment. **But there is one super important thing to watch out for!** When you write a comment it is **required** to leave a space between the `/*` and the first word and the `*/` and the last word. **If you do not use those spaces your script will not work**.

[1] Random Map Script.
[2] The rules we have to follow so the generator can read our script.

```
1 /* this is a comment. */
2
3 /* this is
4 a multi-line
5 comment. */
```

Listing 2: The two types of comments in a random map script. Note the /* and */ which denote the comment's beginning and end. Also note the space between the beginning and end of a comment and the comment itself.

3.2.2 Constants

A constant is a way of binding a *value* to a *name* **permanently** in your script. What kind of value do I mean? Every object and terrain in the game is represented by a *number* that the random map generator understands. To make it so you don't have to memorize a list of weird numbers, the game *predefines* many constants for you.

Definition 3.2. A constant is a number (or a word associated with that number) that the game recognizes as a unit, building, or terrain.

For example instead of typing `1234`, I can type `GRASS` and the game knows that I'm talking about the grassy terrain.

However, some objects and terrains do not come with a predefined name. Instead, we have to look up the number and give it a name of our own. There are a few rules for doing this:

1. The name you choose must not already be assigned to *another number* either by you *or* by the game.
2. The number you choose must be assigned to an actual object in the game. Choosing a random number may cause the game to crash!
3. You define your constant as follows: `#const NAME NUMBER`

Suppose for example, we wanted to include a dead tree in our map. Spooky. We simply do it as shown in Listing 3.

```
1 /* this is how we define a constant! */
2 #const DEADTREE 404
3
4 /* then we can use it like this! */
5 create_object DEADTREE {
6   /* blah blah blah... */
7 }
```

Listing 3: Defining a constant for the dead tree object.

Later in this guide, you will find a section listing all the constants and their predefined names for your convenience.

3.2.3 Sections

A random map script is divided into *seven sections*. Specific parts of your script will go into each section. A section *begins* with the section name enclosed in `<` `>` and *ends* when you start a new one. See Listing 4 for an example.

Definition 3.3. A section is one of the seven segments which separate parts of a random map script.

Note that the sections are shown in a particular order. You *can* place the sections in any order you like. However, **the order in which I have presented the sections is the order in which the game will read them**. In order to avoid confusion, I strongly recommend you use the same order for your sections. This will help you figure why your script is behaving the way it is.

```
1  <PLAYER_SETUP>
2
3  /* player options go here */
4
5
6
7  <LAND_GENERATION>
8
9  /* land generation goes here */
10
11
12
13 <ELEVATION_GENERATION>
14
15 /* elevation settings go here */
16
17
18
19 <CLIFF_GENERATION>
20
21 /* cliff options go here                                    */
22 /* if you don't specify anything, default cliffs will spawn */
23 /* to prevent cliffs, REMOVE this section                   */
24
25
26
27 <TERRAIN_GENERATION>
28
29 /* terrain patches go here                */
30 /* this is usually where you put forests! */
31
32
33
34 <CONNECTION_GENERATION>
35
36 /* create connections like roads between players */
37
38
39
40 <OBJECTS_GENERATION>
41
42 /* spawn units, buildings, resources, eye candy             */
43 /* you can spawn units per player such as TCs and villagers */
```

Listing 4: The seven sections of a random map script.

3.2.4 Blocks

So what do we fill our sections with? *Blocks!*

Definition 3.4. A block is a chunk of code that the generator associates with a specific function.

Each *section* has specific *block types* associated with it. A block type that goes into one section will not work in a different section. For example, the object block you see starting on `line 2` of Listing 5 only works in the objects generation section, which the terrain block in Listing 6 only works in the terrain generation section.

```
1 <OBJECTS_GENERATION>
2 create_object TOWN_CENTER {
3
4   /* ... */
5
6 }
```

Listing 5: An example of an object block.

```
1 <TERRAIN_GENERATION>
2 create_terrain FOREST {
3
4   /* ... */
5
6 }
```

Listing 6: An example of a terrain block.

3.2.4.1 Fields

4 Outline of a Script

4.1 The Preamble

4.2 Player Setup

4.3 Land Generation

4.4 Elevation Generation

4.5 Terrain Generation

4.6 Connection Generation

4.7 Object Generation

4.7.1 Player Towns

4.7.1.1 Regicide Support

4.7.2 Arena Style Maps

4.7.3 Resources

4.7.4 Eye Candy

5 Constants

Information for this section based on Zetnus’ RMS Guide.

The tables are divided into:

1. Buildings
2. Units
   i) Player
   ii) Gaia
3. Resource Objects
4. Terrains/Lands
   i) Normal Terrain
   ii) Forest
   iii) Water

The columns are divided into:

1. Description
   - A brief description of the object/terrain
   - Extra information here

One final important note: **important items will be shown at the top of tables** and remaining items will be sorted alphabetically.

5.1 Buildings

Description	Name	Number
Town Center Use with `set_place_for_every_player`	TOWN_CENTER	109
Wall Use with `set_place_for_every_player` Only works if player lands are connected	WALL STONE_WALL	117
Castle	CASTLE	82
Archery Range	ARCHERY_RANGE	87
Barracks	BARRACKS	12
Blacksmith	BLACKSMITH	103
Bombard Tower	BOMBARD_TOWER	236
Dock	DOCK	45
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,

Description	Name	Number
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,

5.2 Units

5.2.1 Player Units

Description	Name	Number
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,

5.2.2 Gaia Units

Description	Name	Number
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,

5.3 Resource Objects

Description	Name	Number
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,

Description	Name	Number
,	,	,
,	,	,

5.4 Terrains/Lands

5.4.1 Normal Terrain

Description	Name	Number
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,

5.4.2 Forest

Description	Name	Number
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,

5.4.3 Water

Description	Name	Number
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,
,	,	,

6 Example Scripts

6.1 My Script Template

This is the template I use to create all my scripts. I find it very easy to start with this file and edit it to my liking. Clicking the link at the beginning of this paragraph will allow you to copy and paste it.

I have included some extra constants for convenience as well as comments listing popular alternatives to boars, wolves, sheep, forests, etc.

The default settings are for a standard Arabia map. You can change them however you like.

6.2 My GitHub

Check out my GitHub repository which contains all the RMS scripts I have ever written.

Caution: some of these scripts are old and crappy!

6.3 My Steam Workshop

You can also check out my Steam Workshop page.