OpenMW CS Manual Documentation

Release 0.0

HiPhish

January 08, 2017

Table of Contents

The following document is the complete user manual for *OpenMW CS*, the construction set for the OpenMW game engine. It is intended to serve both as an introduction and a reference for the application. Even if you are familiar with modding *The Elder Scrolls III: Morrowind* you should at least read the first few chapters to familiarise yourself with the new interface.

Warning: OpenMW CS is still software in development. The manual does not cover any of its shortcomings, it is written as if everything was working as inteded. Please report any software problems as bugs in the software, not errors in the manual.

CHAPTER 1

Foreword

<Some introductory lines here, explain nomenclature and abbreviations>

1.1 How to read the manual

The manual can be roughly divided into two parts: a tutorial part consisting of the first two (three) chapters and the reference manual. We recommend all readers to work through the tutorials first, there you will be guided through the creation of a fairly simple mod where you can familiarise yourself with the record-based interface. The tutorials are very simple and teach you only what is necessary for the task, each one can be completed in under half an hour. It is strongly recommended to do the tutorials in order.

When you are familiar with the CS in general and want to write your own mods it is time to move on to the reference part of the manual. The reference chapters can be read out of order, each one covers only one topic.

CHAPTER 2

A Tour through OpenMW CS: making a magic ring

In this first chapter we will create a mod that adds a new ring with a simple enchantment to the game. The ring will give its wearer a permanent Night Vision effect while being worn. You don't need prior knowledge about modding Morrowind, but you should be familiar with the game itself. There will be no scripting necessary, we chan achieve everything using just what the base game offers out of the box. Before continuing make sure that OpenMW is properly installed and playable.

2.1 Adding the ring to the game's records

In this first section we will define what our new ring is, what it looks like and what it does. Getting it to work is the first step before we go further.

2.1.1 Starting up OpenMW CS

We will start by launching OpenMW CS, the location of the program depends on your operating system. You will be presented with a dialogue with three options: create a new game, create a new addon, edit a content file.

The first option is for creating an entirely new game, that's not what we want. We want to edit an existing game, so choose the second one. When you save your addon you can use the third option to open it again.

You will be presented with another window where you get to chose the content to edit and the name of your project. We have to chose at least a base game, and optionally a number of other addons we want to depend on. The name of the project is arbitrary, it will be used to identify the addon later in the OpenMW launcher.

Choose Morrowind as your content file and enter *Ring of Night Vision* as the name. We could also chose further content files as dependencies if we wanted to, but for this mod the base game is enough.

Once the addon has been created you will be presented with a table. If you see a blank window rather than a table choose *World* → *Objects* from the menu.

Let's talk about the interface for a second. Every window in OpenMW CS has *panels*, these are often but not always tables. You can close a panel by clicking the small "X" on the title bar of the panel, or you can detach it by either dragging the title bar or clicking the icon with the two windows. A detached panel can be re-attached to a window by dragging it by the title bar on top of the window.

Now let's look at the panel itself: we have a filter text field, a very large table and a status bar. The filter will be very useful when we want to find an entry in the table, but for now it is irrelevant. The table you are looking at contains all objects in the game, these can be items, NPCs, creatures, whatever. Every object is an entry in that table, visible as a row. The columns of the table are the attributes of each object.

Morrowind uses something called a *relational database* for game data. If you are not familiar with the term, it means that every type of thing can be expressed as a *table*: there is a table for objects, a table for enchantments, a table for icons, one for meshes, and so on. Properties of an entry must be simple values, like numbers or text strings. If we want a more complicated property we need to reference an entry from another table. There are a few exceptions to this though, some tables do have subtables. The effects of enchantments are one of those exceptions.

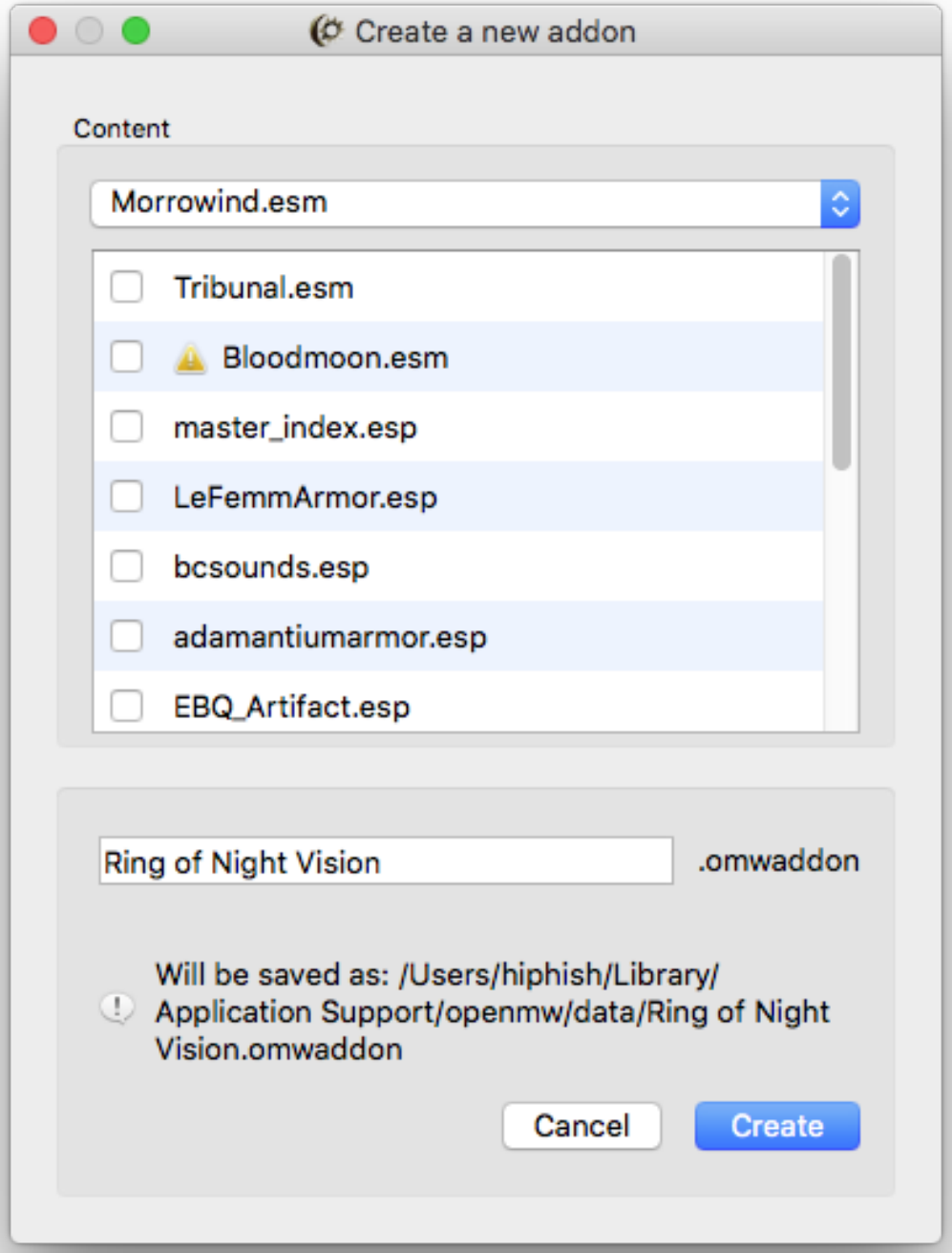
2.1.2 Defining a new record

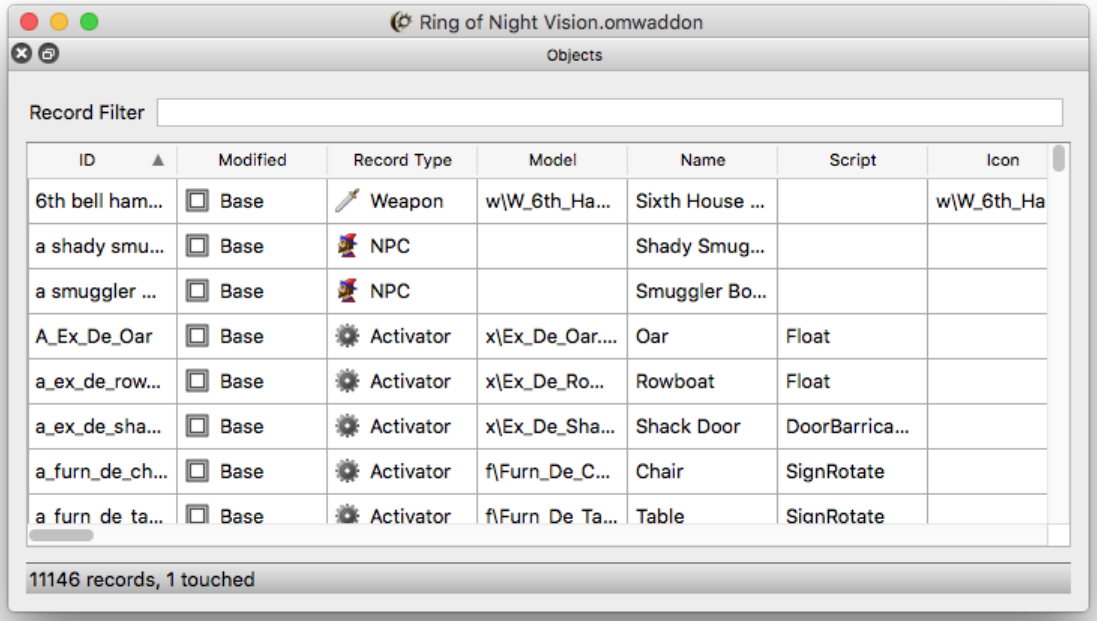
Enough talk, let's create the new ring now. Right-click anywhere in the objects table, choose *Add Record* and the status bar will change into an input field. We need to enter an *ID* (short for *identifier*) and pick the type. The identifier is a unique name by which the ring can later be identified; I have chosen *ring_night_vision*. For the type choose *Clothing*.

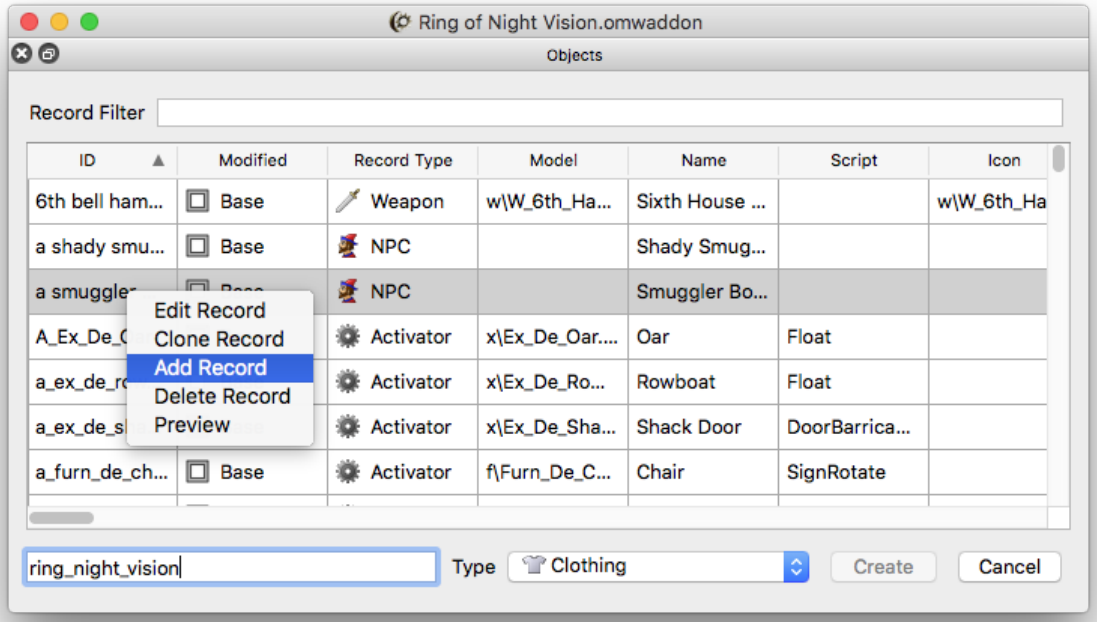
The table should jump right to our newly created record, if not read further below how to use filters to find a record by ID. Notice that the *Modified* column now shows that this record is new. Records can also be *Base* (unmodified), *Modified* and *Deleted*. The other fields are still empty since we created this record from nothing. We can double-click a table cell while holding Shift to edit it (this is a configurable shortcut), but there is a better way: right-click the row of our new record and chose *Edit Record*, a new panel will open.

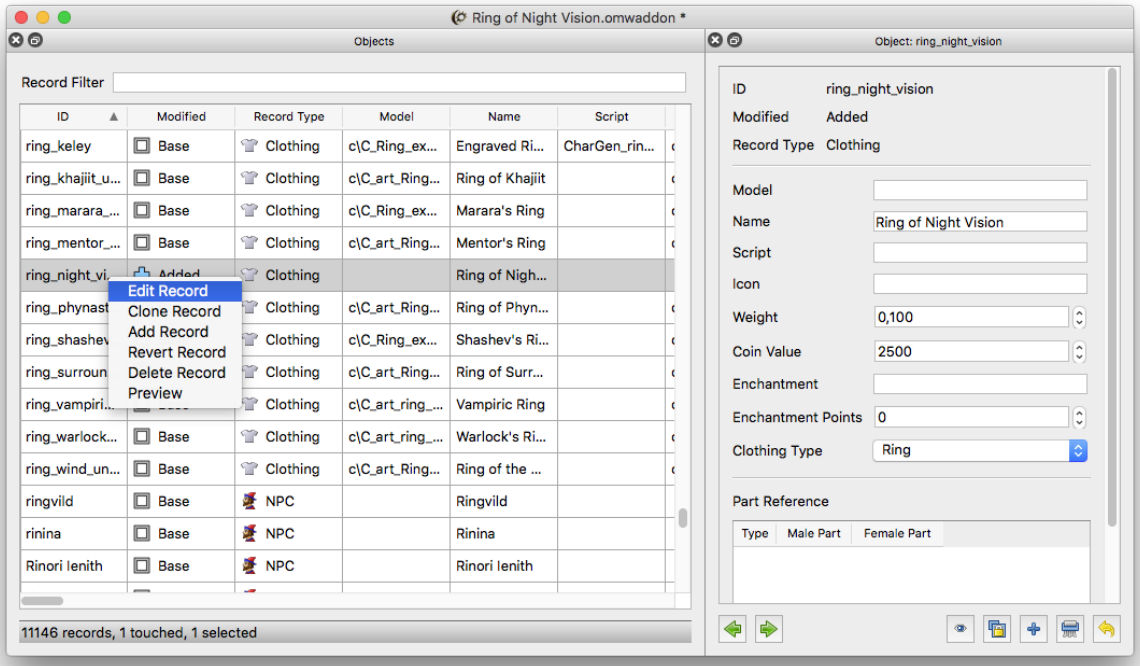
We can right-click the row of our new record and chose *Edit Record*, a new panel will open. Alternatively we can also define a configurable shortcut instead of using the context menu; the default is double-clicking while holding down the shift key.

You can set the name, weight and coin value as you like, I chose *Ring of Night Vision*, *0.1* and *2500* respectively. Make sure you set the *Clothing Type* to *Ring*. We could set the other properties manually as well, but unless you have an exceptional memory for identifiers and never make typos that's not feasible. What we are going to do instead is find the records we want in their respective tables and assign them from there.


Create a new addon
Content
Morrowind.esm
Tribunal.esm
Bloodmoon.esm
master_index.esp
LeFemmArmor.esp
bcsounds.esp
adamantiumarmor.esp
EBQ_Artifact.esp
Ring of Night Vision
.omwaddon
Will be saved as: /Users/hiphish/Library/
Application Support/openmw/data/Ring of Night
Vision.omwaddon
Cancel
Create


Ring of Night Vision.omwaddon
Objects
Record Filter
ID
Modified
Record Type
Model
Name
Script
Icon
6th bell ham...
Base
Weapon
w\W_6th_Ha...
Sixth House ...
w\W_6th_Ha
a shady smu...
Base
NPC
Shady Smug...
a smuggler ...
Base
NPC
Smuggler Bo...
A_Ex_De_Oar
Base
Activator
x\Ex_De_Oar....
Oar
Float
a_ex_de_row...
Base
Activator
x\Ex_De_Ro...
Rowboat
Float
a_ex_de_sha...
Base
Activator
x\Ex_De_Sha...
Shack Door
DoorBarrica...
a_furn_de_ch...
Base
Activator
f\Furn_De_C...
Chair
SignRotate
a_furn_de_ta...
Base
Activator
f\Furn_De_Ta...
Table
SignRotate
11146 records, 1 touched


Ring of Night Vision.omwaddon
Objects
Record Filter
ID
Modified
Record Type
Model
Name
Script
Icon
6th bell ham...
Base
Weapon
w\W_6th_Ha...
Sixth House ...
w\W_6th_Ha
a shady smu...
Base
NPC
Shady Smug...
a smuggle
NPC
Smuggler Bo...
Edit Record
Clone Record
Add Record
Delete Record
Preview
A_Ex_De_
Activator
x\Ex_De_Oar....
Oar
Float
a_ex_de_r
Activator
x\Ex_De_Ro...
Rowboat
Float
a_ex_de_s
Activator
x\Ex_De_Sha...
Shack Door
DoorBarrica...
a_furn_de_ch...
Base
Activator
f\Furn_De_C...
Chair
SignRotate
ring_night_vision
Type
Clothing
Create
Cancel



2.1.3 Finding records using filters

We will add an icon first. Open the *Icons* table the same way you opened the *Objects* table: in the menu click *Assets* → *Icons*. If the window gets too crowded remember that you can detach panels. The table is huge and not every ring icon starts with "ring", so we have to use filters to find what we want.

Filters are a central element of OpenMW CS and a major departure from how the original Morrowind CS was used. In fact, filters are so important that they have their own table as well. We won't be going that far for now though. There are three types of filters: *Project filters* are part of the project and are stored in the project file, *session filter* are only valid until you exit the CS, and finally *instant filter* which are used only once and typed directly into the *Filter* field.

For this tutorial we will only use instant filters. We type the definition of the filter directly into the filter field rather than the name of an existing filter. To signify that we are using an instant filter the have to use *!* as the first character. Type the following into the field:

```
!string("id", ".*ring.*")
```

A filter is defined by a number of *queries* which can be logically linked. For now all that matters is that the *string(<property>, <pattern>)* query will check whether *<propery>* matches *<pattern>*. The pattern is a regular expression, if you don't know about them you should learn their syntax. For now all that matters is that . stands for any character and * stands for any amount, even zero. In other words, we are looking for all entries which have an ID that contains the word "ring" somewhere in it. This is a pretty dumb pattern because it will also match words like "ringmail", but it's good enough for now.

If you have typed the filter definition properly the text should change from red to black and our table will be narrowed down a lot. Browse for an icon you like and drag & drop its table row onto the *Icon* field of our new ring.

That's it, you have now assigned a reference to an entry in another table to the ring entry in the *Objects* table. Repeat the same process for the 3D model, you can find the *Meshes* table under *Assets* → *Meshes*.

2.1.4 Adding the enchantment

Putting everything you have learned so far to practice we can add the final and most important part to our new ring: the enchantment. You know enough to perform the following steps without guidance: Open the *Enchantments* table (*Mechanics* → *Enchantments*) and create a new entry with the ID *Cats Eye*. Edit it so that it has *Constant Effect* enchantment type.

To add an effect to the enchantment right-click the *Magic Effects* table and choose *Add new row*. You can edit the effects by right-clicking their table cells. Set the effect to *NightEye*, range to *Self*, and both magnitudes to *50*. The other properties are irrelevant.

Once you are done add the new enchantment to our ring. That's it, we now have a complete enchanted ring to play with. Let's take it for a test ride.

2.1.5 Playing your new addon

Launch OpenMW and in the launcher under *Data Files* check your addon. Load a game and open the console. We have only defined the ring, but we haven't placed any instance of it anywhere in the game world, so we have to create one. In the console type:

```
player->AddItem "ring_night_vision" 1
```

The part in quotation marks is the ID of our ring, you have to adjust it if you chose a different ID. Exit the console and you should find a new ring in your inventory. Equip it and you will instantly receive the *Night Vision* effect for your character.

2.1.6 Conclusion

In this tutorial we have learned how to create a new addon, what tables are and how to create new records. We have also taken a very brief glimpse at the syntax of filters, a feature you will be using a lot when creating larger mods.

This mod is a pure addition, it does not change any of the existing records. However, if you want to actually present appealing content to the player rather than just offering abstract definitions you will have to change the game's content. In the next tutorial we will learn how to place the ring in the game world so the player can find it legitimately.

2.2 Adding the ring to the game's world

Now that we have defined the ring it is time add it to the game world so the player can find it legitimately. We will add the ring to a merchant, place it in a chest and put it somewhere in plain sight. To this end we will have to actually modify the contents of the game.

2.2.1 Subsection to come...

CHAPTER 3

Files and Directories

In this chapter of the manual we will cover the usage of files and directories by OpenMW CS. Files and directories are file system concepts of your operating system, so we will not be going into specifics about that, we will only focus on what is relevant to OpenMW CS.

3.1 Basics

3.1.1 Directories

OpenMW and OpenMW CS us multiple directories on the file system. First of all there is a *user directory* that holds configuration files and a number of different sub-directories. The location of the user directory is hard-coded into the CS and depends on your operating system.

Operating System	User Dircetory
GNU/Linux	`<whatever>`
OS X	`~/Library/Application Support/openmw/`
Windows	`<whatever>`

In addition to to this single hard-coded directory both OpenMW and OpenMW CS need a place to seek for a actuals data files of the game: textures, 3D models, sounds and record files that store objects in game; dialogues an so one. These files are called *content files*. We support multiple such paths (we call them *data paths*) as specified in the configuration. Usually one data path points to the directory where the original Morrowind game is either installed or unpacked to. You are free to specify as many data paths as you would like, however, there is one special data path that, as described later, which is used to store newly created content files.

3.1.2 Content files

The original Morrowind engine by Bethesda Softworks uses two types of content files: *esm* (master) and *esp* (plugin). The distinction between those two is not clear, and often confusing. One would expect the *esm* (master) file to be used to specify one master, which is then modified by the *esp* plugins. And indeed: this is the basic idea. However, the official expansions were also made as ESM files, even though they could essentially be described as really large plugins, and therefore would rather use *esp* files. There were technical reasons behind this decision – somewhat valid in the case of the original engine, but clearly it is better to create a system that can be used in a more sensible way. OpenMW achieves this with our own content file types.

We support both ESM and ESP files, but in order to make use of new features in OpenMW one should consider using new file types designed with our engine in mind: *game* files and *addon* files, collectively called *content files*.

OpenMW content files

The concepts of *Game* and *Addon* files are somewhat similar to the old concept of *ESM* and *ESP*, but more strictly enforced. It is quite straight-formward: If you want to make new game using OpenMW as the engine (a so called *total conversion*) you should create a game file. If you want to create an addon for an existing game file create an addon file. Nothing else matters; the only distinction you should consider is if your project is about changing another game or creating a new one. Simple as that.

Another simple thing about content files are the extensions: we are using `.omwaddon` for addon files and `.omwgame` for game files.

Morrowind content files

Using our content files is recommended for projects that are intended to used with the OpenMW engine. However, some players might wish to still use the original Morrowind engine. In addition thousands of *ESP*/*ESM* files were created since 2002, some of them with really outstanding content. Because of this OpenMW CS simply has no other choice but to support *ESP*/*ESM* files. If you decid to choose *ESP*/*ESM* file instead of using our own content file types you are most likely aimng at compatibility with the original engine. This subject is covered in it own chapter of this manual.

The actual creation of new files is described in the next chapter. Here we are going to focus only on the details you need to know in order to create your first OpenMW CS file while fully understanding your needs. For now let's jut remember that content files are created inside the user directory in the the `data` subdirectory (that is the one special data directory mentioned earlier).

Dependencies

Since an addon is supposed to change the game it follows that it also depends on the said game to work. We can conceptualise this with an examples: your modification is the changing prize of an iron sword, but what if there is no iron sword in game? That's right: we get nonsense. What you want to do is tie your addon to the files you are changing. Those can be either game files (for example when making an expansion island for a game) or other addon files (making a house on said island). Obviously It is a good idea to be dependent only on files that are really changed in your addon, but sadly there is no other way to achieve this than knowing what you want to do. Again, please remember that this section of the manual does not cover creating the content files – it is only a theoretical introduction to the subject. For now just keep in mind that dependencies exist, and is up to you to decide whether your content file should depend on other content files.

Game files are not intend to have any dependencies for a very simple reasons: the player is using only one game file (excluding original and the dirty ESP/ESM system) at a time and therefore no game file can depend on other game file, and since a game file makes the base for addon files it can not depend on addon files.

Project files

Project files act as containers for data not used by the OpenMW game engine itself, but still useful for OpenMW CS. The shining example of this data category are without doubt record filters (described in a later chapter of the manual). As a mod author you probably do not need or want to distribute project files at all, they are meant to be used only by you and your team.

As you would imagine, project files make sense only in combination with actual content files. In fact, each time you start to work on new content file and a project file was not found, one will be created. The extensio of project files is `.project`. The whole name of the project file is the whole name of the content file with appended extension. For instance a `swords.omwaddon` file is associated with a `swords.omwaddon.project` file.

Project files are stored inside the user directory, in the `projects` subdirectory. This is the path location for both freshly created project files, and a place where OpenMW CS looks for already existing files.

3.1.3 Resource files

Unless we are talking about a fully text based game, like Zork or Rogue, one would expect that a video game is using some media files: 3D models with textures, images acting as icons, sounds and anything else. Since content files, no matter whether they are *ESP*, *ESM* or new OpenMW file type, do not contain any of those, it is clear that they have to be delivered with a different file. It is also clear that this, let's call it "resources file", has to be supported by the engine. Without code handling those files it is nothing more than a mathematical abstraction – something, that lacks meaning for human beings. Therefore this section must cover ways to add resources files to your content file, and point out what is supported. We are going to do just that. Later, you will learn how to make use of those files in your content.

Audio

OpenMW uses FFmpeg for audio playback, and so we support every audio type supported by that library. This makes a huge list. Below is only small portion of the supported file types.

mp3 (MPEG-1 Part 3 Layer 3) A popular audio file format and de facto standard for storing audio. Used by the Morrowind game.

ogg An open source, multimedia container file using a high quality Vorbis audio codec. Recommended.

Video

Video As in the case of audio files, we are using FFmepg to decode video files. The list of supported files is long, we will cover only the most significant.

bik Videos used by the original Morrowind game.

mp4 A multimedia container which use more advanced codecs (MPEG-4 Parts 2, 3 and 10) with a better audio and video compression rate, but also requiring more CPU intensive decoding – this makes it probably less suited for storing sounds in computer games, but good for videos.

webm A new, shiny and open source video format with excellent compression. It needs quite a lot of processing power to be decoded, but since game logic is not running during cutscenes we can recommend it for use with OpenMW.

ogv Alternative, open source container using Theora codec for video and Vorbis for audio.

Textures and images

The original Morrowind game uses *DDS* and *TGA* files for all kinds of two dimensional images and textures alike. In addition, the engine supported *BMP* files for some reason (*BMP* is a terrible format for a video game). We also support an extended set of image files – including *JPEG* and *PNG*. *JPEG* and *PNG* files can be useful in some cases, for instance a *JPEG* file is a valid option for skybox texture and *PNG* can useful for masks. However, please keep in mind that *JPEG* images can grow to large sizes quickly and are not the best option with a DirectX rendering backend. You probably still want to use *DDS* files for textures.

CHAPTER 4

OpenMW CS Starting Dialog

In this chapter we will cover starting up OpenMW CS and the starting interface. Start the CS the way intended for your operating system and you will be presented with window and three main buttons and a small button with a wrench-icon. The wrench will open the configuration dialog which we will cover later. The three main buttons are the following:

Create A New Game Choose this option if you want to create an original game that does not depend on any other content files. The distinction between game and addon in the original Morrowind engine was somewhat blurry, but OpenMW is very strict about it: regardless of how large your addon is, if it depends on another content file it is not an original game.

Create A New Addon Choose this option if you want to create an extension to an existing game. An addon can depend on other addons as well optionally, but it *must* depend on a game.

Edit A Content File Choose this option is you wish to edit an existing content file, regardless of whether it is a game or an addon.

Whether you create a game or an addon, a data file and a project file will be generated for you in you user directory.

You will have to choose a name for your content file and if you chose to create an addon you will also have to chose a number of dependencies. You have to choose exactly one game and you can choose an arbitrary amount of addon dependencies. For the sake of simplicity and maintainability choose only the addons you actually want to depend on. Also keep in mind that your dependencies might have dependencies of their own, you have to depend on those as well. If one of your dependencies nees something it will be indicated by a warning sign and automatically include its dependencies when you choose it.

If you want to edit an existing content file you will be presented with a similar dialog, except you don't get to choose a file name (because you are editing files that already exist).