

# Creating Mods for ECWolf

## Part 1

- Working with Maps
- MAPINFO scripts

# File Formats

- While ECWolf supports the original file formats (WL6), these formats are not recommended for distribution and do not contain enough information for ECWolf to work with on their own.
- Instead we'll be using standard ZIP files as well as the WAD format from Doom.
  - You may use the WAD format for the whole project, but it is generally agreed that ZIPs are easier to work with.
  - Be sure to use the PK3 extension when distributing your mod in order to signal to the user that the file should be loaded by the game and not extracted!

# Tools (PK3)

- Slade 3  
<http://slade.mancubus.net/>
  - This tool will be used for managing your PK3 file.
  - To be supplemented by a map editor.
- Technically any zip archiving tool will work, but Slade handles file format conversions.



# Tools (Mapping)

- One of the following two editors are usable for ECWolf.
  - WDC  
<http://winwolf3d.dugtrio17.com/>
  - Havok's Wolf Editor  
<http://hwolf3d.dugtrio17.com/index.php?section=hwe>
    - If you choose to use HWE, you will need to use the feature to export in WDC's format.
- Use these editors only for map editing. Slade can handle editing all of the other resources.

# Tools (Mapping)

- ChaosEdit and Others

<http://www.chaos-software.de.vu/>

- Other editors may be used with ECWolf as well, but they can not export the map in the correct format.
- You may use these editors during development, but you will need to use one that support WDC's format when you are ready for release!

# Mapping

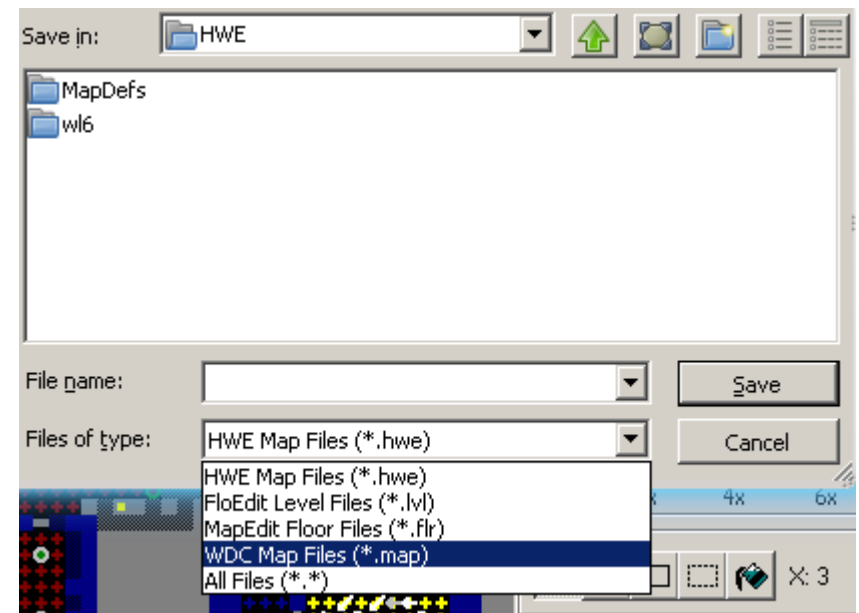
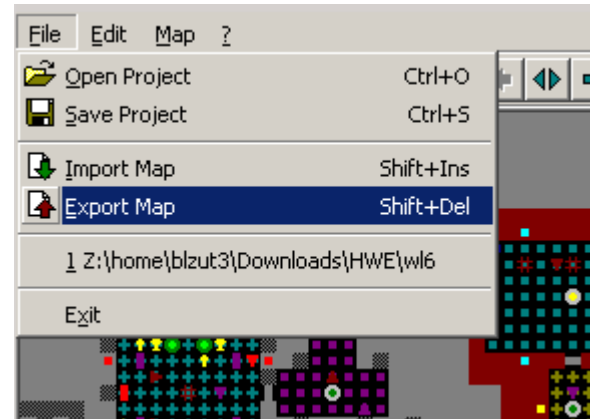
- ECWolf supports two map formats. Natively a new textual map format called UWMF exists, but as of this writing there are no editors.
- Because of this, binary format maps can be used.
  - During development you may load your gamemaps.wl6 with ECWolf directly. This will assign maps to the slots MAP01-MAPxy (where xy=number of maps).
  - For public release you will want to export and package your map into your PK3.

# Exporting Maps (WDC)

- View the map you want to export
- Click “Import/Export”
- Click “Export to File”

# Exporting Maps (HWE)

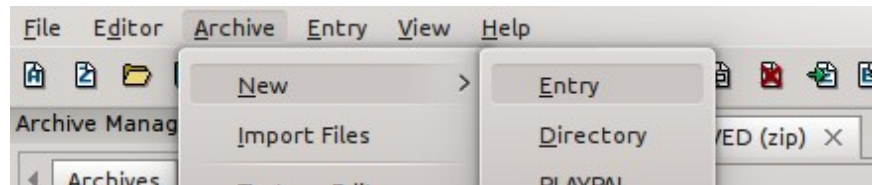
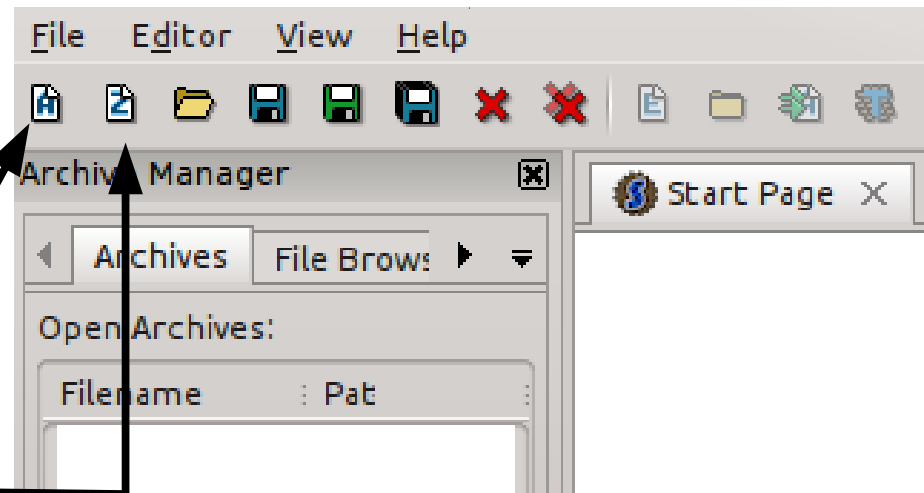
- File->Export Map
- Click “No” to export in different map format.
- Select “WDC Map Files”
- Select which maps to export





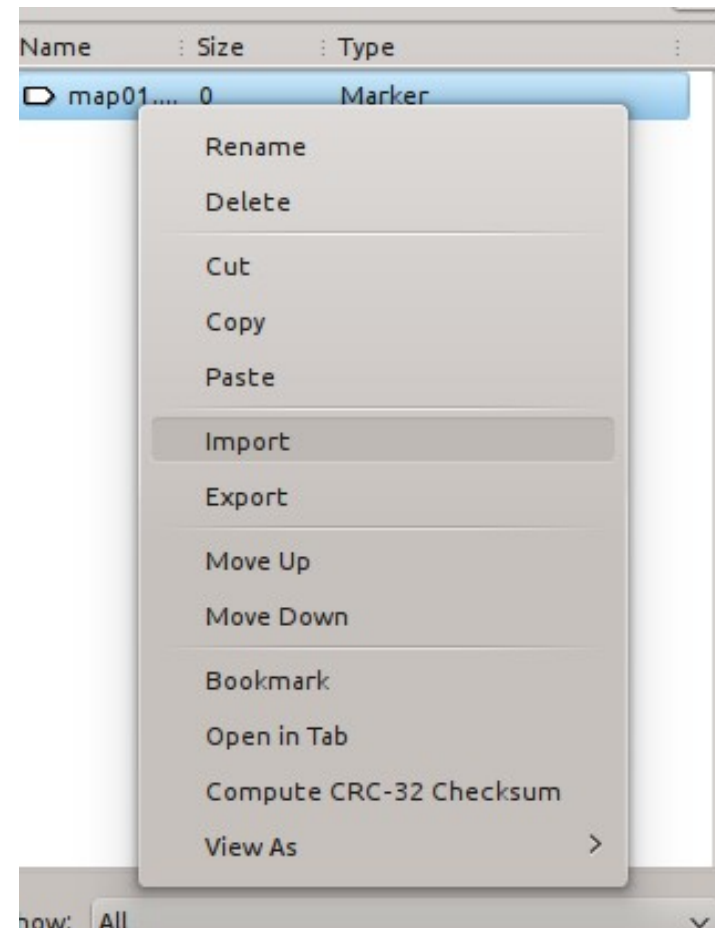
# Getting Started

- Create a few maps in your editor of choice and export
- Now use Slade to do the following:
  - Create and save an empty wad (we'll use this later!)
  - Create a new PK3
  - Create a new entry called mapinfo.txt
  - Create a directory called maps



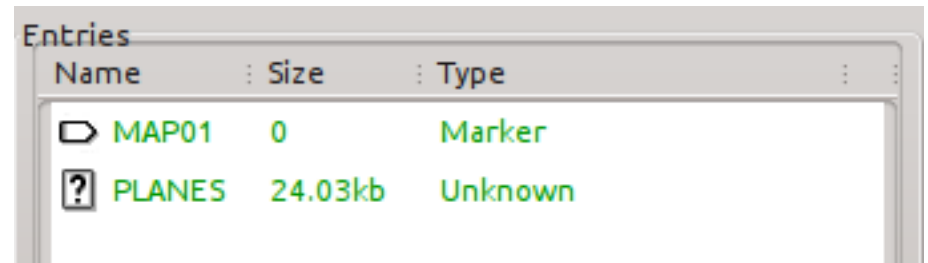
# Importing Your Map

- Go into the maps/ directory
- Create a new entry, map01.wad
- With this entry, import your empty wad file
- Double click to open it



# Importing Your Map

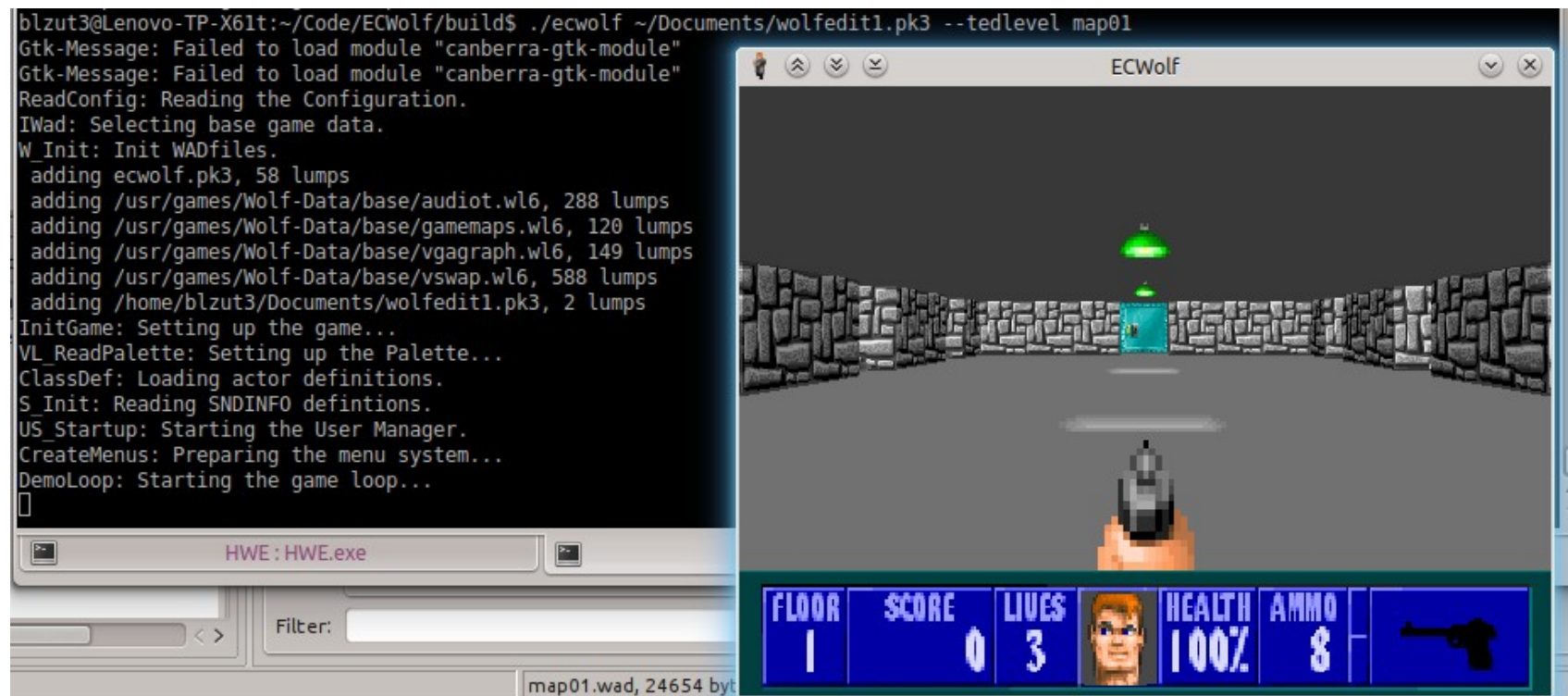
- Create two entries (in this order!): MAP01 and PLANES
- Leave MAP01 empty, but import your exported map into the PLANES entry
- Save and close



Entries			
Name	Size	Type	
MAP01	0	Marker	
PLANES	24.03kb	Unknown	

# Importing Your Map

- Before we look at the mapinfo file, we can try out our map as-is!
  - Either drag and drop, or use the command line to load the file. With the latter `-tedlevel` can warp to the map.



# Scripting Basics

- Before diving into scripting we should learn a few basic terms universal to all scripts
- It is possible to learn to use most, if not all, of ECWolf's scripting formats while skipping the formalities, but you will have trouble with reading the error messages
- **Comments**
  - ECWolf uses C-style comments for all of its scripting formats
    - `//` This is a single line comment
    - `/*` This is a block comment `*/`

# Scripting Basics

- **Identifier**

- Any unquoted string of alphanumeric (a-z, 0-9) characters or underscores. Identifiers, however, may not start with a number
- Valid identifiers: MyActor, `_some_thing_`, key2
- Invalid identifiers: 2000pointgiver

# Scripting Basics

- **String Constant**

- Any series of characters enclosed in quotes (“”).
  - Example: “This is a string constant.”
- Strings may contain quotation characters, but only if escaped. Escaping means it is preceded by a backslash (\). Likewise backslashes need to be escaped.
  - Example: “This \"statement\" contains quotes and this is a backslash: \\”

# Scripting Basics

- **Integer Constant**

- Any series of numeric numbers (0-9)
  - Examples: 0, 5, 25, 125
- Technically an integer constant is a positive number without decimals although in general negative numbers are accepted as well.

- **Float Constant**

- Integer constant with decimals allowed
  - Examples: 1.5, 0.25, 25.0
- Can contain scientific notation as well, but use of this form is uncommon



# Scripting Basics

- **Blocks**

- Code enclosed within braces ( { } )
- It is highly recommended to indent your code, either by a tab or a few spaces, when entering a block. Doing so will help prevent obvious parser errors.
- Examples:
  - ```
if( ... ) {  
    // Some prefer opening brace on same line  
}
```
  - ```
else  
{  
    // Others put braces on their own line  
}
```

# Scripting Basics

- **Scope**

- Closely related to blocks is the concept of scopes. For now you should be aware of the concept of “global scope.”
- If something is said to be in the global scope, this simply means it is not inside any block and stands on its own.

# Intro To MAPINFO

- <http://maniacsvault.net/ecwolf/wiki/MAPINFO>
- The mapinfo, despite what the name should leave you to believe, is the configuration file for the game
  - This includes configuring episodes, play classes, global settings, and more
- To start with, we'll focus on maps, but it is important to be aware that there are other things that can be set with it!

# MAPINFO Syntax

- The mapinfo file follows this general syntax:

```
blockheader ...  
{  
    key = value, value2, value3  
    flag  
}
```
- Keys and flags are identifiers
  - The number of comma separated values the key accepts depends on the key. Refer to the wiki for what the values are.
- Flags are lone identifiers that take no values

# Map Block

- There are technically three forms of the map block header. For now we'll look at the most basic form. The others can be found on the wiki.
- Header:
  - map “MAP01” { /\* Properties \*/ }

Block name

MAP Label

# Map Block

- Refer to the wiki for a complete list of properties:

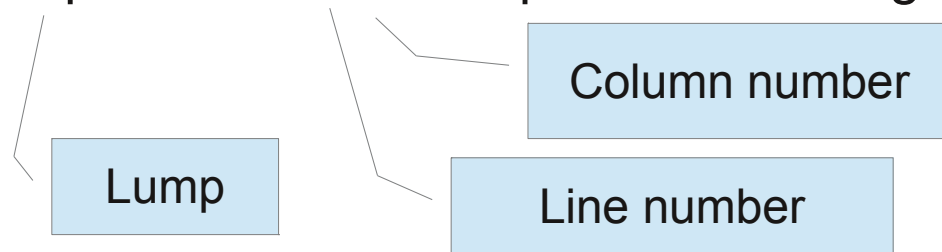
[http://maniacsvault.net/ecwolf/wiki/MAPINFO/Map\\_definition](http://maniacsvault.net/ecwolf/wiki/MAPINFO/Map_definition)

- Example:

```
map "MAP01" {  
    // Set really ugly floor/ceiling colors  
    defaultfloor = "#800000"  
    defaultceiling = "#000080"  
    // Set the music  
    music = "POW"  
    // Set the next map  
    next = "MAP02"  
    // If this was next = "EndTitle" then it would end the episode  
}
```

# Errors

- If there is an error in your script file, ECWolf will print an error to the terminal. These error will point to the line and column on which the parser could not continue. This does NOT necessarily mean that the error is at that point, just the the computer could not continue execution.
- Format:  
mapinfo.txt:12:4:Excepted 'Identifier' got 'String Constant' instead.



- In this case this would mean that on line twelve there was a string where a normal identifier should be

# Lump Names?

- As you may have noticed ECWolf assigns a lump name to every chunk in the vanilla data, but you may be wondering what these names are?
- Open `ecwolf.pk3` and check out `wl6map.txt` (or `sodmap.txt` if editing for Spear of Destiny)
  - For each block, every string corresponds sequentially to the chunks of that type.
- You may notice there's `sd2map` and `sd3map`. ECWolf gives unique names to all of the mission pack resources, so they may be used simultaneously in mods!




# Episode Block

- Now that we know how to edit the progression and appearance of maps, you might want to also edit the episode list
- To do this, we will usually want to clear the original episodes leaving just our own. This can be done by adding “clearepisodes” to the global scope
- One thing to note is we need at least one episode to operate. If ECWolf only finds one episode, then it will skip the episode selection menu

# Episode Block

- Wiki reference:  
[http://maniacsvault.net/ecwolf/wiki/MAPINFO/Episode\\_definition](http://maniacsvault.net/ecwolf/wiki/MAPINFO/Episode_definition)
- An episode is a pointer to the starting map. ECWolf does not track which episode the player selected, so they may freely overlap if you wish
- Header:  

```
episode "MAP01" { /* Properties */ }
```



Starting Map

# Episode Block

- Example:

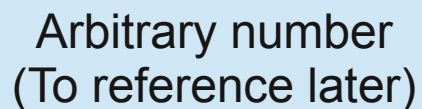
```
clear episodes
episode "MAP01" {
    name = "My First Mod"
}
```
- Do remember that if you do not define another episode the menu will NOT appear!

# Cluster Block

- There is one more part to the level progression, clusters
  - A cluster is a group of maps. The transition between which may contain a story sequence
- Wiki reference:  
[http://maniacsvault.net/ecwolf/wiki/MAPINFO/Cluster\\_definition](http://maniacsvault.net/ecwolf/wiki/MAPINFO/Cluster_definition)

- Header:

```
cluster 1 { /* Properties */ }
```



Arbitrary number  
(To reference later)

# Cluster Block

- Example:

```
cluster 1 {  
    exittext = "Hello World"  
    // We can also reference lumps ("art files")  
}
```

- But we also need to reference the cluster in the map!

```
Map "MAP01" {  
    /* Properties as before */  
    cluster = 1  
}
```

- Now when we transition to a map in another cluster or EndTitle, the exit text will be displayed.

# Conclusion

- This concludes our starter on mapping in ECWolf
- Please refer to the Wiki for a complete reference on mapinfo properties
- The next presentation will cover the basics of working with actors