Oblivion Mod Maker's Manual

Compiled by Lord_Gannondorf
Lord_Gannondorf@hotmail.com
V1.4

The Elder Scrolls Construction Set

Version 1.2.404

Disclaimer

I am not in any way affiliated with Bethesda Softworks, ZeniMax, or any company or person who works for these companies. None of these companies in any way allowed or prevented me from completing this work, and most likely were unaware of its production prior to it being provided over the Internet. Nor did any of these companies or anyone who works for them in any way provide any input or help me in compiling this work.

This work is provided freely, and may be downloaded, printed, saved, or distributed to others through any means. However, you may not alter this work, nor profit in any way, if you distribute this work or provide it for distribution, without the express permission of myself.

This work is currently version 1.4 and refers to the Construction Set version 1.2.404. There is still some information within that is incomplete, or unknowingly inaccurate or false. I take no responsibility for anything any person does with their computer system due to the information contained within, nor any problems that may arise at any point in the future. Use this work at your own risk.

If you see information contained within that you originally provided and wish it removed or changed for accuracy, or wish to be credited, please contact me and I will do so immediately.

Finally, I am not soliciting for help, but if anyone sees an inaccuracy or wishes of their own will to send me information to improve this work, I will gladly accept it. If it is included in this manual I will include their name, or the identity they provide, in the Credits section. If more than one person sends me the same information, I will credit the one to first send it to me, or all who do so concurrently.

-Lord_Gannondorf

Lord_Gannondorf@hotmail.com

Table of Contents

Chapter 3 - Toolbar Description....28

Chapter 4 - Windows....46

Introduction

Before I begin, I would like to point out that most of this work in this manual has been copied out of the Morrowind Mod Makers Manual, Complied by Edwardsmd. But, it wasn't as simple as it sounds, quite a bit has changed in the Oblivion Editor from the Morrowind Editor, I had to read through all that I copied and make sure it was the same as in Oblivion, if it wasn't the same; I had to change it.

Also, some of this manual I have taken from TESCS Wiki and other various sources. Check Websites for the link to the Wiki.

Quite a small update this time; I have added The Sigil Stone Window, The Potion Window, The Subspace Window and have updated and added various other sections.

I will be updating this manual as regularly as I can; if you find anything incorrect or false or that you originally provided and wish it removed or changed for accuracy please contact me.

Thanks for downloading and reading.

Welcome to the world of modding.

-Lord_Gannondorf

Credits

The credits are listed in no particular order. If you find some information in this manual that you originally wrote and want credit please contact me and I will put your name here.

Edwardsmd – He compiled the Morrowind Mod Makers Manual (MMMM). If the MMMM didn't exist either would this! Thanks to him and everybody who contributed information to the manual, you are all very great people.

Logam – For writing his Modding Guide, I '*borrowed*' a few pieces out of it.

Everybody at TESCS Wiki – Without this, modders would be lost. Thanks to *everybody* that contributed information to this website.

Povuholo – For writing the 'making a quest tutorial' included in this manual.

Koroush Ghazi – I used his tweaking guide for Oblivion in this manual.

Sativarg – He converted the manual into PDF format and submitted various information throughout the manual.

Abbreviations and Definitions

.7z - A compressed file, like a .zip file.

.ace - A compressed file, like a .zip file.

Acrobat Reader - A free program that can be used to view any .pdf file.

Adobe - Company that created PhotoShop, Acrobat, and Acrobat Viewer.

AI – "Artificial Intelligence" – the part of the program, which controls the behavior of "living beings" in the game.

3D Studio Max - Professional program used to create the 3D objects of the game.

.bsa - This file holds all the objects, sounds, etc files used by an .esm.

Bug - A programming error. Just about any program contains some minor "bugs", which can be fixed with updates or patches.

CS (or TESCS) – The Elder Scrolls Construction Set.

CTD - Crash To Desktop; when you're playing Oblivion and it suddenly stops, disappears, and is replaced by the desktop.

D-Click, D-Click'd - Double click.

.dds – Image format used for textures in Oblivion.

Dependent – Plug-in files are dependent upon one or more master files. You cannot load a plug-in in the CS, nor in the game, without loading the master files it is dependent upon.

Discreet - Company that owns and sells 3D Studio Max.

.esm - Elder Scrolls Master file; like a plugin only you cannot modify it.

.esp - Elder Scrolls Plug-in file. All changes made with the CS are stored in this file.

.ess - Elder Scrolls Save file. These are your saved game files.

Expansions – Knights of the Nine (KOTN) and Shivering Isles (SI) are the two official expansions available for Oblivion.

FPS - Frames Per Second. Your computer redraws the screen constantly. Each time it does this is referred to as a 'frame'. The more it has to draw the fewer frames per second.

Less than 30 FPS is considered observable by the human eye.

GIMP - A free software program used to create ‘image’, or ‘art’, files (i.e. textures).

HD or HDD - Hard Drive, or Hard Disk Drive.

ID - Identification (name of an object or thing in the CS). Not necessarily the *Name* of an object, but the *ID name* the CS keeps track of an object by.

.kf - File associated with a .nif file that has animation in it.

L-Click, L-Click’d - Left-click.

Leveled List - A list created with creatures or treasure. It is accessed by the game to provide a creature to fight appropriate to the PC’s level, or distribute treasure appropriate to the PC’s level.

Meshes - The 3-dimensional wireframe form of an object.

MilkShape - A cheap program used to create meshes. Not as capable as 3D Studio Max.

Mod - A plug-in, to include all files necessary to play it.

.nif - NetImmerse File Format. All 3D models in Oblivion are this format.

Northmarker - Object placed in an interior cell to denote which way is north.

NPC - Non Player Character, every character in the game except the one playing.

OBSE – Oblivion Script Extender

Object - Anything that can be placed in the game, is an object, and treated as such by the game.

Paint Shop Pro - Software used to create ‘image’, or ‘art’, files.

PC - Player Character, the one playing the game.

.pdf - Portable Data File. This type of file can be opened and viewed by anyone on a PC or MAC computer with Adobe Acrobat Reader.

PhotoShop - Software used to create ‘image’, or ‘art’, files.

R-Click, R-Click’d - Right-click.

.rar - A compressed file, like a .zip file.

Readme.txt - A simple text file that explains what a file is for, what it does, how to install it, and what to expect. This is basically everything about the file. Should be included with any mod you create, upload, or download.

Reference - Any time you place an object in the game you create a *reference* of that object.

RPG - Abbreviation of "role-playing game".

Script, Scripting - Manually written code to execute, or augment, an effect in the game (often hidden from the player, though it's effects or results may not be).

Skills - Statistical actions you can take to accomplish something in the game.

Spell - A magical effect cast by actors/PC composed of spell effects.

Spell Effects - Spell effects are used to create spells that are used in the game world.

Skin - The texture file that is 'wrapped' around a mesh to create an object.

Skinning - The process of altering and fitting a texture onto a wireframe to create an object.

Stats - The PC's defining statistical abilities (attributes and skills).

TC - Total Conversion; when someone completely deletes everything from Oblivion (all cells, landscape, references, etc) except the game engine, and creates a new "game" from scratch as an .esm file.

TES - The Elder Scrolls.

TESCS (or CS) - The Elders Scrolls Construction Set.

Textures - An 'image' file used to 'skin' a mesh to create an object.

Toggle - Usually a key, or combination of keys, that produces an affect, or stops it, by 'turning it on' or 'off' with repeated keystrokes.

XP - Common abbreviation for "Experience Points".

.zip – A compressed file.

Shortkeys

Many shortkeys in the CS do not need a key combination to work (i.e. you do not need to hit ctrl + ‘key’ to accomplish something). Just hit the shortkey to toggle or activate a command or response within the CS.

Render Window shortkeys:

A - Toggles on/off brightness with the Render Window.

B - Toggles on/off yellow borderlines delineating cells.

C - Centers your viewpoint upon an object at ground level, facing north.

D - Deselects your viewpoint from an object.

F – Drops an object to the ground or object underneath it.

H - Toggles on/off the Landscape Editor.

L - Toggles on/off viewing the radius of light source objects.

T - Centers your viewpoint above a selected object.

S - When held, click on an object to resize it.

V - When held, allows you to zoom in or out by moving your mouse.

W - Toggle on/off wireframe mode.

X - When held, allows you to move an object with your mouse on the X-axis.

Y - When held, allows you to move an object with your mouse on the Y-axis.

Z - When held, allows you to move an object with your mouse on the Z-axis.

F4 - Shows collision detection of all objects within the Render Window.

‘**Shift**’ - When held, allows you to rotate, or spin, your viewpoint 360 degrees with your mouse (if an object is selected, you will rotate around that object).

‘**Spacebar**’ - When held, allows you to change your viewpoint upon the XY-plane (sideways), or the Z-plane (up and down) with you mouse.

‘**Mousewheel**’ - Allows you to zoom in or out, if you have one.

Chapter 1 - TESCS – What is it?

With the CS you can create new items, NPCs, spells, races, birthsigns, towns, dungeons, classes, factions (i.e. guilds), quests, or alter what's provided with the game. There are still some things you cannot do with the CS, and there are some things you *should not* do with it. When working with the CS, keep in mind the reason it was provided: to *extend* the life of Oblivion as a product for *sale*. Period. The designers of the game may feel some sense of altruism in releasing the CS, but you're praying for a jackalope sighting if you attach the same to those who published the game. They did it to increase *profit*. With this in mind realize you cannot create your own "game" with the CS, there are built in limitations. Otherwise, why buy "their game", or a "future game" from them when you can make your own. So when you want to cry because you can't create your own .esm file (we'll get to this), or wonder why you can't create new Skills, Attributes, or Spell Effects, new or different 'bodies' for Races and curse and kick at the shortsightedness of those who created/limited the Scripting commands and functions, relax; just enjoy the game, and have fun modding.

How a plug-in works:

When you work on your plug-in, or mod, you are not actually changing anything in the Oblivion.esm file. You are making changes stored as an .esp file. When the game is loaded, the game loads an .esm (or more than one), and then loads any plug-ins. The game then makes changes based upon each plug-in as it is loaded, in order. What this means is, you cannot do anything that can ruin the game itself. If a plug-in causes problems, just exit the game, open up the Data Files menu, uncheck the plug-in(s) causing problems, and click Play. Master files cannot be affected in any way.

What you can do with the *TESCS:*

What you are able to do is quite a lot, but can be summarized this way:

- You can alter almost everything within the game.
- You can add new objects (weapons, spells, races, land, towns, dungeons, etc) by re-using what is provided within the CS, or creating new ones based off of what is provided.
- You can modify objects within the CS to do something different.
- You can modify the way the game world operates and how the player interacts with it.
- Create and/or modify an .esp file.

What you cannot do with the *TESCS*:

---

[1]

What you cannot accomplish is actually quite small, but very absolute. This is done to protect the stability of the game, and enable mods produced by different people with different ideas to work together if installed, and loaded in a game at the same time. Also to protect the profit line of the game. You cannot:

- Create new meshes or textures for items or races in the game.
- Create/delete spell effects used to create spells within the CS.
- Create or delete skills or attributes.
- Make changes to the physics engine (though you can alter some aspects of how it operates).
- Create or alter an .esm file.
- Create or alter an .ess file.

Note: *Scripting enables you to accomplish things the designers may not have thought of, or get around inherent limitations of the game. This was intentional, but scripting also has a great many limitations to prevent abuse, maintain stability, and to keep the spirit of the game in line with what is stated in the above bullets.*

What is an ESM/ESP/ESS/INI file?

These are the files that hold all the information, or data, the game uses to run. In simple terms:

- **.ESM** (Elder Scrolls Master) files are the **master files** that contain all the data to play the game; besides the models, textures or sound. The Oblivion.exe file holds the data on how the game functions. This keeps things simple, and also why unlike other games you don't need to install a 2.0 GB file on your hard drive to play. You cannot alter, change, modify, or delete anything in an .esm file with the CS. Even if you try. Why did Bethesda do that? So we can't screw the game up and spend all our time on the phone with Bethesda's customer service bitching and complaining about their stupid game that we screwed up.

- **.ESP** (Elder Scrolls Plug-in) files are the **plug-in files** that we create when we save changes we have made to the game. We can alter, modify, add, and delete these files to our late-night, bleary-eyed, caffeine-induced heart's content. Then post it on the Internet for others to try out. You can even do this to someone else's .esp file you've downloaded from the Internet. Some few have done this to modify and improve a good mod that just needed a few tweaks to make it a great mod. Of course, some post dirty/buggy mods too. Thankfully, this is somewhat rare.

- **.ESS** (Elder Scrolls Save) files are your **saved game files**. Lots of people like to make saved game editors to cheat and boost their character's abilities for games that are published, especially RPGs. I have even seen some for Oblivion. Why? With the CS you can already do this outside of gameplay, or with the console in game. You shouldn't have any reason to mess with this file.

- **Oblivion.ini** This specific file provides data for the game to start up and run. I don't suggest you mess with this file unless you know ***exactly*** what you are doing. However, there are tweaks you can do to this file to improve or fix your game if it has trouble running properly. Some mods also force you to make minor changes to it for them to run properly. If you wish to make changes to this file, ***make a back up copy first.*** If you screw it up or your changes cause problems you can always replace it with an unchanged copy.

If you want to know what any other file is, don't worry about it. You don't need to know. You'll most likely screw something up anyway, so just leave everything else alone unless you have obtained a screw-up-your-computer-but-good-and-still-be-able-to-fix-it-college-degree.

So then, what the heck are these nif, kf, dds, bsa, etc. for?

You just have to know, don't you? Like me you can't leave well enough alone, you just have to know. Fair enough. I'll do my best to explain each:

.bsa – These files have all the sounds, meshes and textures stored in them. These files can be quite large; there are also utilities created by several skilled programmers that enable you to open the .bsa and view, extract, or add files to it.

.dds - The type of art file used as a texture for a mesh. You need to download plugins to modify or make .dds files in various programs.

.kf - This file comes along with a .nif file for animation purposes.

.mp3 - This is a compressed audio file that provides music and voices for the game.

.nif - This file is the mesh for all physical objects in the game. Texture files are associated/linked with it. All texture files are .dds. If an object has animation then a .kf file usually will be associated with a .nif. Some .nif files may also have a second .nif file that holds additional data for its animation (such as an activator).

Opening Multiple Construction Set windows:

Yes, you can do this. Simply open up the `oblivion.ini` file and add 'AllowMultipleEditors=1' somewhere in the general section on a line by itself. Now you can open multiple windows of the CS at one time.

```
   Oblivion.ini Directory:

My Documents\My Games\Oblivion\Oblivion.ini
```

Note: *Be advised, the CS is not the most stable program ever created. I do not recommend having extra multiple Construction Sets open for more than reference to a master file's or mod's contents. Going crazy with this has resulted in crashing some of the CS windows on my system. But that may just be me.*

Enabling screen shots:

Dying to get some screenshots posted of your mod? Open up the Oblivion.ini, in the Display section, you will find bAllowScreenShot=0. Change the '0' to a '1'. Now when playing in game, when you want to take a screen shot, just tap the `Print Screen` button. A screenshot will be created in your `Oblivion` folder on your Hard Drive, in `.bmp` format.

```
   Oblivion.ini Directory:

   My Documents\My Games\Oblivion\Oblivion.ini
```

Note: *Screenshots cannot be taken if you have Anti-Aliasing enabled. Be sure to disable this in your Video settings before you try to take screenshots.*

How do I load files into the CS?

When you load files into the CS it works in a similar way to starting a game with a plug-in. Click any .esp files you want loaded, you may click more than one. Highlight the one you want *active*. Any changes made to any .esp files loaded will all be saved to the *active* .esp. Now, you need to keep something in mind here; if you make a *change* to an .esp that is loaded but is not the *active* .esp, you might just have made **your** .esp *dependent*. If you load an .esm file that an active .esp is not dependent on, and at any point click the Save button, the .esp file will become dependent upon that .esm plus any other .esm files it is already dependent on. Keep this in mind when choosing what you want to load and base your .esp off of. A cautionary note here, you do not have to place a check next to an .esm. For when loading an .esp, the game will load any .esm files an .esp is dependent on. However, I have generated errors when doings this at times (no clue why, and it doesn't happen often or with all plug-ins). I suggest you always mark what you want to load. If you don't want to load any .esp files, just .esm files, check the box next to those you wish to load. There is no need to make one active; the active button is only for .esp files. The .esm files always load first, then .esp files that are marked. The .esp files are listed in the

Data Files window from top to bottom, by date (oldest date at the top, newest at the bottom). The date stamp on an .esp is when last Saved.

I don't have KOTN/SI, but the mod I downloaded needs it/them to run!

Sorry, but you will have to go buy them.

Can I download the TESCS, Oblivion, KOTN, and/or SI from the Internet?

You can only download the CS from: http://www.elderscrolls.com/, but downloading the expansions or Oblivion itself is Illegal.

I bought Oblivion, so it's legal for me to use a no-CD crack, right?

Wrong. Use of a no-CD crack is illegal, period.

What about Horse Armor and other downloadable content?

Well, you can download them from the internet but they do cost money, although you can get all but one of the downloadable content by purchasing KOTN; they are included on the CD. (The Fighters Stronghold is the mod not included, why? Because KOTN was released before Fighters Stronghold.)

What TESCS version have I got?

First of all, you should always have the latest version of the CS. At the moment the latest version of the CS is 1.2.404. You want to know what version you have, click on the Help Menu on the toolbar, and then click *about*. This will bring up the window that is at the beginning of this manual. The window has your version number right up the top.

I have bought KOTN, but the quest wont start!

This is a bug in KOTN, sometimes the quest begins sometimes it doesn't. To start the quest simply open the console with the tilde key (~) and type the following: SetStage ndpilgrim 0.

Chapter 2 - Installing/Playing Mods

Where do you get all the best mods? You can probably get them from the same place you got this reference manual. Once you have the mod you most likely will have take steps to unzip it, possibly place files where they belong, and then you can play it. Maybe. First, let’s figure out what file format the mod you downloaded is in.

Note: *A very simple way to find mods is to go to google and search for:*
Elder Scrolls IV: Oblivion Mods.

The file is a .rar, 7z, ace, zip. What do I do?

Ah yes, I remember the first time I downloaded a file, saw the .rar extension, and said to myself, “What is this? How do I use it?” Well, fear not o’ ignorant one. It’s just a different form of file compression, as is an .ace or.zip format. Open either one and if you have a utility that can open them, you’re in business. If Windows brings up a little window asking you to choose the program to use with this file, you need a utility or program to unzip these mods. Almost any compression utility will open a .zip file, some will open a .rar, and others will open an .ace. Here is a breakdown for you:

These will open a .zip file:
-WinRar
-ZipGenius
-WinZip
-PKZip
-7zip

These will open a .rar file:
-WinRar
-ZipGenius
-7zip

These will open an .ace file:
-WinRar
-WinAce
-ZipGenius
-7zip

These will open a .7z file:
-WinRar
-7zip
-ZipGenius

Note: *You get what you pay for is a good rule of thumb here. I tried several different ‘free’ utilities that kept screwing files up when I tried to extract them. Maybe it was the*

-

program; maybe it was just my system. As you see WinRar opens many different file types, it is certainly one of the better programs.

Where do I put all these files?

Most people include a readme.txt file with their mod (*and a pox upon those that don't!*), so read it, that's what it's for. They should tell you where to put the files they included. In general, the files should always go somewhere within the Data subfolder within the Oblivion folder (most of the time). Some modders will produce a .zip file that will *self-extract*. This means all you have to do is open or D-Click it. The file will do the rest and you're done. Most files you will be prompted to choose where to extract the files. The readme.txt provided is the source for this info, but most likely it will be the Data subfolder. Most compression utilities have the ability to open a readme.txt while still compressed. Just open the compressed file and look for the readme.txt. Double click it, and it should open for your reading pleasure. If you are leery of unzipping straight into your Oblivion subfolders (and I don't blame you), just extract to a temporary folder to examine the contents, and then you can manually move them over. I don't recommend this though. If you don't know what you are doing, or are not conversant with file placement, you could mess something up. Again, I ***highly*** recommend you do what the readme.txt tells you to do.

Nif & kf - These are meshes/animations, and they always go within the Meshes subfolder.

Dds - These are textures; they go within the Textures subfolder.

Mp3 & wav – These are sounds, put it in the Sound subfolder.

Esp - Put these in the Data folder.

Readme.txt - Don't delete these! Keep them; you might need to refer to them again (and again and again and…). I keep mine all within the Data folder. Make sure the name isn't just 'readme.txt', rename it to include the name of the plug-in so you know which one it is for.

There is an added hitch; the modder may have put his files in a separate subfolder within one of the above-mentioned folders, just so their files are easy to find. You can't always tell this by looking at the contents of a compressed file. This is why I recommend you extract to a temporary file, at a minimum, before you install a plug-in. An easy way to install a plug-in is to extract to a temp folder. Once extracted look and see what came out. If you find some folders labeled Meshes, Textures, Sounds, etc, you're in luck. Drag and drop these folders into the Data Files folder, click 'yes' (it should be ok to overwrite), and that's it.

-

But what if you extract and all you get is a slew of different file types, not even a folder to hold them. Most often seen as a modders resource, in which case you need to figure it out. If you can figure out where they go, go for it. If not, it probably won't hurt anything if you put something in the wrong place, but the plug-in probably won't work correctly (or maybe not at all). If you are not familiar with these file types, and are not entirely sure what each file extracted is exactly for and where it goes, don't mess with it. Ask someone, such as the author of the plug-in, or post on a forum for help. I know how much fun it is to uninstall and/or reinstall a game or backed up file, but I really don't recommend it.

No readme.txt included?

Delete it! If you don't know what the different files are that Oblivion uses, and where to put them, you don't know enough to be messing around like this. You might accidentally replace a needed file, or put something in the wrong place.

How do I play this mod?

When Oblivion starts up and you're staring at the autorun screen there is a button that says **Data Files**, click it. Just place a check mark next to any mods you want to play. Make sure to also check KOTN and/or SI if they're necessary to play one of the mods you load. Once done click 'Ok' and click on 'Play'. Enjoy!

Plug-in conflicts:

If two plug-ins are loaded that conflict you will have problems (and you might not know it until somewhere along the story line). There are utilities to check for conflicts, while not perfect, they are very, very, good. Something that usually doesn't provide conflicts but can still cause problems is *Leveled Lists*. Many plug-ins make use of leveled lists, but only those leveled lists in the last plug-in loaded are used by the game! Leveled lists from other plug-ins are ignored or overwritten! While this might not render a loaded mod unplayable, it generally will lower or limit the game play value. To fix this you can use a utility that merges these leveled lists and allows you to load this singular master leveled list as a plug-in; this is very useful.

Where's the Console?

To open the console all you have to do is hit the "~", or 'tilde' key as its known. On most keyboards it is the only key to the left of your '1' key. If not there, just look for it. On non-English keyboards you should be able to use the key to the left of the '1' key, regardless of what it is. A couple things to know though when using the console; you can open the console regardless of whether you have opened up your menus or not (i.e. right-clicked). Right clicking with the console open, but the menus closed will resume game play, and the console will remain open until you hit the 'tilde' key again. Mess with it a little bit to see how it works. This can be useful; if the console is open and you resume play you will notice that when you 'look' or center the screen upon any object, that object’s ID is displayed in the game (not in the console, but in the game). This can be invaluable. Here some console functions and there effects:

Function Name	Short Name	Description	Parameters
CenterOnCell	COC	Move player to specified interior cell	CellName
CenterOnExterior	COE	Move player to specified exterior cell coordinates	x, y
CenterOnWorld	COW	Move player to specified cell in specified Worldspace [COW worldname -10 5]	WorldspaceName, CellCoordX, CellCoordY
CompleteAllQuestStages	CAQS	Sets all quest stages to finish	
LoadGame	LOAD	LoadGame <filename>	SaveName
MoveToQuestTarget	MOVETOQT	Move player to current quest target (optional param: target number, 1 to N).	QuestTargetNumber (optional)
PickRefByID	PRID	Select a reference by id for the console.	ObjectRefID
PlayerSpellBook	PSB	Add all spells to player.	
PrintAiList	PAI	Printed Ai Lists.	
PrintHDRParam	PHP	Prints current HDR settings.	
PrintNPCDialog	PDIALOG	Prints NPC dialog	
PurgeCellBuffers	PCB	Forcibly unloads all unattached cells in cell buffers.	
QuitGame	QQQ	Exit game without going through menus.	
RefreshINI	REFINI	Refresh INI settings from file.	
RefreshShaders		Reload HLSL shaders from disk	
ReloadCurrentClimate	RCC	Reloads values from the current climate	
ReloadCurrentWeather	RCW	Reloads values from the current weather	

SaveGame	SAVE	SaveGame <filename>	SaveName
SaveIniFiles	SAVEINI	Writes all the .ini files.	
SaveWorld		Save hkWorld <filename>	Filename
SetCameraFOV	FOV	Change the camera's field of view (in deg): default 75	iDegrees
SetClipDist		Float, new clip distance	fClipDistance
SetDebugText	SDT	Sets what debug text is shown.	iDebugPage
SetFog		2 floats, start and end depths	fStartDepth, fEndDepth
SetGameSetting	SETGS		sGSName, sGSValue
SetHDRParam	SHP	Sets various values for the HDR shader	fEyeAdaptSpeed, fEmissiveHDRMult, fTreeDimmer, fGrassDimmer, fValue5, fValue6
SetINISetting	SetINI		sININame, sINIValue
SexChange		Selected npc male become female or female becomes male.	
Show	TST	Show values of script variables: show gamedayspassed	VarName
Show1stPerson	S1ST	Show the 1st person Model from the 3rd person camera. If in 3rd person mode it will show both.	
ShowAnim	SA	Show Animation and Actor status.	
ShowFullQuestLog	SFQL	Show all log entries for a single quest	QuestID
ShowQuestLog	SQL	Show Quest Log. Optional flag: 0=current quests, 1=completed quests	bShowCompleted
ShowQuests	SQ	List quests.	
ShowQuestTargets	SQT	Show current quest targets	

ShowQuestVars	SQV	Show quest variables. [Sqv QuestID]	QuestID
ShowRenderPasses	SRP	Display render passes for the next frame	
ShowSubtitle		Show all dialog subtitles (1 shows always,0 hides always)	bFlag
ShowVars	SV	Show variables on object. [player->sv]	
ShowWhoDetectsPlayer	SWDP	Show who detects the player	
StartAllQuests	SAQ	Starts all quests	
TestAllCells	TAC	Test All Cells (0 - stop, 1 - start, 2 - interiors, 3 - current world)	iValue
ToggleAI	TAI		
ToggleAiSchedules	TAIS		
ToggleBorders	TB	Show borderlines for each cell.	
ToggleCastShadows	TSH		iShadowType
ToggleCharControllerShape	TCCS	Toggle char controller shape type.	
ToggleCollision	TCL		
ToggleCollisionGeometry	TCG	Show collision geometry.	
ToggleCombatAI	TCAI	Toggles ALL Combat AI	
ToggleCombatStats	TCS		
ToggleConversations	TCONV	Toggle conversation stats	
ToggleDebugText	TDT	Shows framerate (FPS) and debug numbers on the screen.	
ToggleDetection	TDETECT		
ToggleFlyCam	TFC	Toggles the Free Fly camera (UFO cam).	
ToggleFogOfWar	TFOW	Turns fog of war on or off.	
ToggleGodMode	TGM	Toggle God mode	
ToggleGrass	TG	Toggle grass display.	
ToggleGrassUpdate	TGU		
ToggleLeaves	TLV		

ToggleLiteBrite	TLB	Toggles lite brite render mode.	
ToggleLODLand	TLL		
ToggleMagicStats	TMS		
ToggleMapMarkers	TMM	Toggle map markers (1 shows all, 0 hides all).	iValue
ToggleMaterialGeometry	TMG	Show material geometry.	
ToggleMenus	TM	Hide all the menus. Used for taking screen shots.	
TogglePathGrid	TPG	Toggle blocked display.	
TogglePathLine	TPL	Toggle path display.	
ToggleRefractionDebug	TRD	Toggles refraction debug render texture	
ToggleSafeZone	TSZ	Display the television 85% safe zone	
ToggleScripts	TSCR	Turn Script processing on/off	
ToggleShadowVolumes	TSV		
ToggleSky	TS		
ToggleTrees	TT	Turn trees on/off	
ToggleWaterRadius	TWR		
ToggleWaterSystem	TWS	Toggles the water system	
ToggleWireframe	TWF	Show the world as wireframe.	

Use these commands at your discretion. Many are useful for when trying to debug something you’ve done in your mod, but doesn’t work, or at least not the way you want it to. Some I’m not sure what they are supposed to do, or what it is they are doing. Some of the Toggle commands are useful for testing how the game responds to changes you’ve made with a mod, others most likely were useful for the developers, but provide little more than confusion or amusement now, and probably should have been, but were not, removed when the game was released.

Lastly, remember that many commands are not listed here that can be used in the console window. You can add/delete objects and spells, modify skills and attributes, resurrect NPCs if killed but shouldn’t have been, and in general make many modifications to the game. For those who like to cheat, the console is easy and available. There is also nothing wrong with cheating, to each his own. But if you like to mod, you will find the console invaluable in game.

I have found a bug in my mod!

Bugs. We all hate them. It's inevitable that they will show up, and the more complex and bigger your mod, the more likely you will have them (and more of them!). How do you get rid of them? The simple answer is you have to figure out what is not working right, go back into your plug-in in the CS, figure out what you did wrong, and fix it. There is no simple way to do it either; you just have to do it. But there are a number of things you can do to simplify this process:

Write it down!

First and foremost; write down every single addition, change, or deletion you make to your mod. Sounds like a lot of trouble right? It is, and it will draw the modding process out timewise, believe me. So what, do it anyway. I have caught bugs before I even hit the save button by looking over my notes as I work. Get yourself a notebook, or one with sections. Each page or section can pertain to changes you have made to the world pertaining to an area, or to an aspect of the game (like spells, skills, races, etc). Be meticulous, note everything you do. If you write down you modified an object, write down its' name **AND** it's ID, exactly as it's spelled. Put down where it is in the world; the cell, outside or inside what building/dungeon, on what floor, on top of what, etc. Added a script to it? Write it down. Shifted this thingy around to make room for said object? Write it down. Make a mistake and dirtied something up? Write it down! You will have to clean that out. Being able to look over your notes and catching problems before you save a mod is a real stress reducer. A lot easier than spending untold amounts of time in-game playing your mod before you come across the bug, and then pulling your hair out trying to figure out what you did wrong. Your notes also give you the basic draft for writing your readme.txt to include with your mod.

Make new objects!

Don't make changes to objects you find in the game, make a new object! When you make a change to an object in the CS, regardless of whether it is an object reference in the Render Window, or you click on a parent object in the Objects Window, any change you make to that object affects every single instance of that object everywhere in the game! This can't be stressed enough! If you do this you might just be screwing the game up somehow so it can't be played or completed. Or possibly rendering your mod completely incompatible with almost every single other mod created that's out there.

Don't make a dirty mod!

If you click on an object in the game to modify it in some way, then change it back to it's prior state of existence, the CS notes that you changed that object in some way and stores that information in your mod! The CS can not distinguish between an actual change you make to something, or the fact that you changed something back to the way it was, only that an object has been in some way altered! This probably won't screw up your mod (notice I said probably), but what it does do is add size to your mod, making it bigger. A few of these won't add noticeably to a plug-in's size, but if you are one of those who doesn't care about this, you probably have many such dirty changes. Which means there's a greater chance your mod is both buggy, and much bigger in size than necessary, which means a lot longer time for download (What? You think everyone is running a DSL line to his or her computer?).

Resist the urge to create uber mods!

When you first open the CS it's tempting to create an uber-item so you can waltz through the game, and a lot of people do at some point. I did it (after my first dance through the game), and it was fun; for a while. Then the enjoyment wore off. But some people seem to think their uber-item is so powerful and cool everyone should have it too, and they upload it for the masses, joining the ranks of uber-items for download. Most websites now have enough of these they don't need more. There are also plenty of tweak mods out there. What are these? Mods that change the entire aspect of the game in almost every way to make it harder, or easier. These are really great and take a great deal of work, and lots of knowledge and effort. These are a real challenge to create. But there are plenty, why create what someone else already has? If you can make one in a new and unique way, or different from others, I say go for it! Originality is something the mod community loves to see. But do you want put effort into something, only to upload it and find out someone else did the same thing some time ago? I'm not trying to dissuade anyone from attempting anything. Do what you want and like. It's up to you. But I like to plumb the depths of unexplored territory, or at least do something in a way no one has. But that's just me.

Use forums on websites to get help!

Can't figure out how to do something? Can't figure out why something isn't working the way you want? Want some advise or a mod tested by someone else for insight? Join the forum on a website you like, and post questions, troubles, or trawl for some advise there. You might find out some things you didn't know, or possibly someone already asked your question, and got an answer (which you need). People on forums are usually, reasonably, friendly. Almost always helpful, and you might be helping someone else out too. This is a personal choice, and up to you. I myself gleaned enormous amounts of useful information that was invaluable from forums at several websites, and usually got exactly the help I needed when I posted a question asking for it. You just about can't go wrong.

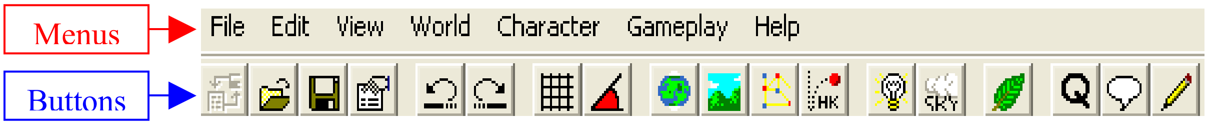
Can I play all 5-gigazillion mods I installed?

The short answer, yes. The long answer, most likely yes. The game's actual limit is 255 files loaded at one time. That's both .esp and .esm files. I've seen people state they have loaded and are running over 200 plug-ins; it is possible. However, the ability of your system to handle the plug-ins you load, regardless of how many, is the real limitation. Some people create mods that really test your machine's ability to churn out that FPS. A slower machine might have trouble with some mods that add lots of scripts to be executed globally, constantly (everyone seems to like using global scripts and variables to work around the game's limitations. Nothing cuts the FPS like globals). Or have too many objects animated and interacting all at the same time on screen. Having a ton of plug-ins loaded and playing at the same time could produce these conditions. The real answer is not how many can you have loaded on your machine, but which ones can you load that your machine can handle together.

Chapter 3 - Toolbar Description

The Toolbar:

This is the toolbar. From here your mod is given life, your imagination will become virtual reality, and your life will be shortened by 1.7 years due to stress. And that is the point of this manual, to reduce or eliminate that 1.7 years of life lost from stress. So let's start with the actual Toolbar and see what we can find to do on it.

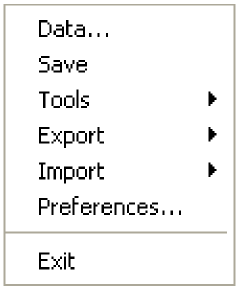


The

The toolbar is not exactly the average windows toolbar, which is quickly obvious if you click on any of the menus to see what drops down. I will discuss each menu and each button in order. Those menu functions that open a window I will go into greater depth later in the manual. Many of the menu listings are also accessible by clicking one of the buttons, and many also have a shortcut key combo. Let's start with the menus.

The Menus:

File Menu:



-

File->Data:

This will open a window that allows you to view all plug-ins (.esp) and master files (.esm) installed on your computer. Master files are listed first. Then plug-ins will be listed, with the last one listed having the newest date. To load an .esm or .esp simply double click it, and a check mark appears next to it. This means it will get loaded into the CS when you hit the "ok" button at the bottom. Some things to remember:

When you load a plug-in, any .esm that it is dependent on will also be loaded, regardless of whether or not you marked it for loading. This is automatic and you cannot prevent it. This is true even when loading multiple plug- ins with different master file dependencies. All necessary master files will be loaded.

TES files:

Pick what plug-in listed you want to load. Once you've made your decision on what to load, you must highlight one file and click the "Set as Active File" button. This plug-in will be considered *active*. This plug-in will be the only one you make changes to that can be saved. You can still see data from any other plug-ins that you also load; you just cannot save any data to those plug-ins. Once chosen just click the 'ok' button and the CS will load any checked plug-ins and master files. If you load only master files you do not need to make one active, the CS will load the `.esm` files and any changes you make will become a new plug-in (you will be prompted to name it if you hit save). If you forget to set a plug-in as active, the CS will bring up a prompt stating this and ask if you wish to continue. If you click "No", the CS will close. If you click "Yes" the CS will load up and when you save it will be dependent upon that .esp.

Created By:

Who created this file? You can enter your name, or ID, in this block.

Summary:

Here you can write a small summary about your plug-in. If it's small with few changes you probably don't need to say much. You should at a minimum list what this plug-in is about, and a version number.

Parent Masters:

Here you will find listed all master files that a plug-in is dependent upon.

Details:

This button brings up a window that allows you to see all the changes you've made to your mod. The window lists all data within a plug-in. If you see something here you don't remember creating or doesn't have any effect on your mod, your mod is dirty and needs to be cleaned.

File->Save

Click and save your plug-in. I have noticed that when working within some windows (Dialog, Scripting, and Cell Path Grid) if you click the Save button, then close the window, the changes you make are saved only within that aspect of the game. You must still click the save button to save it to your plug-in once that window is closed.

File->Tools->Combine Loaded Plug-ins

Load any and all plug-ins you want combined. Click on this option under the File menu. A prompt will appear asking if this is what you want to do. If so, a window will appear and you will have to provide a name for the new combined plug-in that will be created. Do so and hit save. Your new combined plug-in will become the loaded plug-in, displacing all others, become the active file, and will appear at the bottom of the list in the Data window. Consider this option carefully before you do this, combining plug-ins that conflict will produce problems.

File->Tools->Create Difference Plugin

I tried making a difference plugin with several different ways and got nothing but an empty ESP. I think it's just a placeholder for a function that is not yet implemented or not included in the public release.

File->Tools-> Create Filtered Difference Plugin

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

File->Tools-> Filter File By ID's

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

File->Tools-> Copy Plug-Ins Audio Files

As the name suggests, once you have your plugin loaded click this and a new window will appear; this lets you select the source directory for the audio files. Once you select a folder and click *ok* the target directory window will appear. Simply click on a new folder to copy the audio files there or create a new folder by clicking the Make New Folder button at the bottom of the window and the files will be placed in there.

File->Tools-> Resave With New ID's

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

File->Tools-> Update Plugin Audio File ID's

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

File->Tools-> Create Map to Map Map

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

File->Tools-> Apply ID Map To Altered Forms

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

File->Import/Export

When I saw these menu items I was hoping for the ability to use text files to import data, and/or to export data to another plug-in. Not quite. Sometimes I have gotten it to export data and then import it to another plug-in, other times it doesn't. Haven't figured out yet why, suspect it only works when data for an object is already present, and you are merely making changes. It will not work to add (or delete) objects to an `.esp`.

File->Preferences

This will bring up a window with options to change the rate at which you can move objects within the Render window, move your view within the object window, load cells as you work, and what you can view of the exterior/interiors (The Preferences window can also be displayed by clicking the preferences button on the toolbar; to the right of the save button)

Movement Tab

Snap to Grid.

When the Snap to Grid is checked, objects in the Render Window will move a number of units determined by the number enter in the Grid Snap box. The CS states to use multiples of 8 (8, 16, 24, 32, etc), though any number can be used. This can be of use in moving objects quickly and getting them to line up with other objects easier.

Note: *This is useful when placing large static objects, like when putting together a building (interior or exterior). But for small objects it's probably easier to do manually without this enabled. Really it's a matter of choice, what you feel works best for you.*

Snap to Angle.

When the Snap to Angle is checked, objects in the Render Window will rotate a number of degrees determined by the number entered in the Snap to Angle box. Remember there are 360 degrees in a circle. See above ***Note*** about use.

Snap to Reference.

This is very handy. If you click Select Reference in Render Window you can click on an object to snap another piece to it. This is done by selecting the other piece and activating grid snap. Very handy for snapping building pieces together!

Movement Speeds:

These are the speeds at which objects or the viewpoint moves or rotates in the Render Window.

Landscape Movement, Sensitivity Multiplier:

This is the rate at which the landscape mesh/wireframe is changed, or moves, in response to your mouse when landscaping. The default is quite slow, good for small-refined changes. Increase if you wish to make large changes quicker.

PathGrid Connection, Auto Distance:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Render Window Tab

Time of Day:

This makes the sky look different if you have the sky displayed.

Clipping Distance:

In simple terms, effectively, this is how far you can view into the distance. Move the slider bar closer to near for slower systems for better and/or quicker response from the CS. There is a single tick mark approximately ¼ over from the left on the bar. This is exactly 8192 units in viewable distance (the size of a cell).

Note: *I like have it set close (for a single cell) when adding/deleting or modifying objects in a cell. Then about halfway when adding buildings and structures for a town or village that spans more than one cell. When landscaping I set it to max and use wireframe mode. I change this slider quite often as I work. Use what works for you.*

Allow Render Window Cell Loads.

When unchecked, the CS will only load cells, and their objects, that are visible in the Render Window. Cells not fully visible will not be loaded; wilderness cells will not be created outside of the Render Window. If checked, the CS will load visible cells (even partially visible ones), and a two cell 'buffer zone' around each and every fully, or partially, visible cell in the Render Window. This may slow down some slower systems.

Shaders Tab

Don't even mess with this. Unless you fully understand what this does, leave it alone. I messed around with it and it stuffed up the CS. Trust me, if you don't know what it does, Leave it!

LOD Tab

These control the distance that you can see Actors, Items or/and Objects in the CS. Making them lower will make less objects appear further away; very handy for low-end machines.

Misc Tab

Skip Initial Cell Load On Editor Start:

When a plug-in is loaded cell data loaded for cell 0, 0. This prevents that from happening. I have found unchecking this to have a minimal effect in load time for a plug-in, but it may help on slower systems.

Skip Initial User Plugin Load:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Auto-Save:

Check this box if you prefer the CS to auto-save your plug-ins, and enter how often in the minutes box. I have this feature off to prevent saving something stupid I didn't realize I did making my mod dirty or unplayable. I also hit the save button after every single change I make out of habit and experience.

Preview Movement Tab:

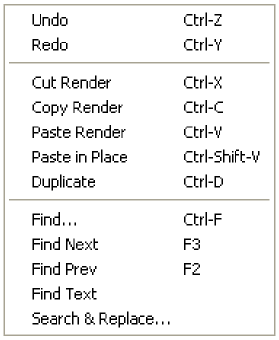
Preview Movement:

These are the speeds at which objects or the viewpoint moves or rotates in the

Preview Window.

File->Exit. Quit the CS.

Edit Menu:



Edit->Undo/Redo:

Undo a mistake you made, or redo something you clicked undo on, but decide to keep.

Edit->Cut/Copy/Paste Render:

This is your standard cut, copy, and paste. It only applies to objects in the Render Window.

Note: *You can use the mouse pointer to box in multiple objects, thus selecting them as a 'group' and treating them as such for cut/copy/paste.*

Edit->Paste in Place:

This will place cut/copied objects at the selected spot, upon the ground or floor.

Edit->Duplicate:

Clicking this will create a duplicate of selected object(s). The duplicate will appear exactly on top of the object(s) duplicated.

Edit->Find:

Brings up the find window. This window has a drop down menu listing of all objects in the plug-in. You choose which one you wish to find and it searches references until it finds one. It then shows the object's reference in the Render Window in the cell it found it in. You can use the next set of menu options to see the next or previous found reference. If an object has no references in the game, it will not execute.

Edit->Find Next/Previous.

This will cycle through the objects references 'found' with the Find option.

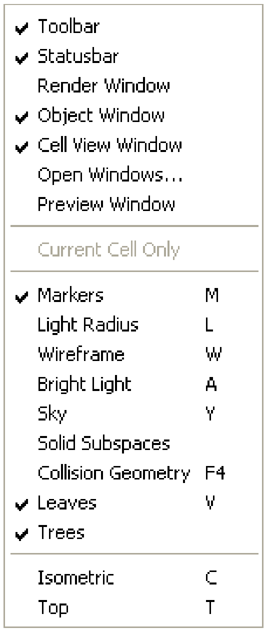
Edit->Find Text:

This option brings up a window with a search box, a button to the right of it labeled ‘Find text’ to start the search, and three tabs. These tabs are labeled Dialog, Script, and Object. It will list all dialog, script, and objects in the appropriate tabbed window according to your search criteria. The function appears to accept results that may not be clearly related to your search criteria (though usually are).

Edit->Search & Replace:

This brings up a small window with two drop down boxes. The first is the object you wish the CS to find all references for, and the second is the object you want the first to be replaced with. There are two check boxes. One will execute the function in the current cell; the other will only operate on a selected object (which should be listed in the first drop box) or group of objects.

View Menu:



View->Toolbar:

This will display or not display the buttons beneath the menus.

View->Statusbar:

This will display or not display the bar at the bottom of the CS window displaying information on currently selected objects and actions.

View->Render Window:

This will display or not display the Render Window.

View->Object Window:

This will display or not display the Object Window.

View->Cell View Window:

This will display or not display the Cell View Window.

View->Open Windows

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

View->Preview Window

This previews the objects in the game.

View->Current Cell Only:

Completely grayed out. I have no idea what it does, if anything, and could not get it to be active.

View->Markers:

This will display or not display North, Door, Temple, Divine, and Travel markers in the Render Window.

View->Light Radius:

This will display or not display the radius of light emitted by an object by the presence of a globe encircling an object that emits light.

View->Wireframe:

This will display or not display the landscape and all objects in wireframe mode.

View->Bright Light:

This will brighten everything within the Render Window. By default the Render Window is somewhat dark. To better see objects within it, click this option. Just keep in mind it will look different when in-game; if you forget to put any lights in because you ‘brightened’ your Render Window…

View->Sky:

This displays the sky in the render window. Refer to The Buttons->Sky.

View->Solid Subspaces:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

View->Collision Geometry:

This will display or not display the collision grid in the Render Window, showing where the PC will collide with objects. This is only for objects, not landscape.

View->Leaves:

This will display or not display the leaves on trees, take this off if you have a low-end machine. It increases performance considerably.

View->Trees:

This will display or not display trees; this can also increase performance considerably.

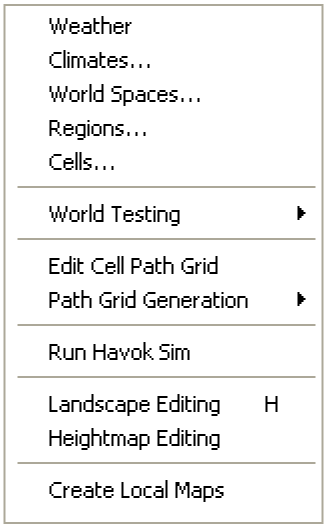
View->Isometric:

This will center your viewpoint upon a selected object at ground level, facing north

View->Top:

This will center your viewpoint upon a selected object above said object, looking straight down at it (north is up on your screen).

World Menu:



World->Weather:

This window controls all the weather settings. You can modify any of them or make your own weather effects. You can adjust anything with the weather in here; you can adjust the wind speed, the sky textures, lighting and heaps more.

World->Climates:

This window is basically the same as the weather window, except it controls the climates.

World->World Spaces

This is where you can create a whole new Tamriel! In here you can make a new world space by right clicking in the grid to the left and clicking new. You name you new world whether it is a town or island, or even an Oblivion Plane. You can change the Music, Water, and Climate or even add a map!

World->Regions:

This allows you to change the settings for regions, including weather and sounds. A region is a collection of cells that are of similar make (i.e. the Great Forest, Jerall Mountains, etc), but they don't *have* to be similar. A town is part of a region, and a single cell, or several cells, could be quite different yet still be considered part of a region.

World->Cells

This brings up a window to create an interior cell. It does not create anything within the cell, it only allows one to adjust the lighting, allow sleep, set a water level, or, believe it or not, create an interior cell and set it to act like an exterior cell. Refer to the Cells Window for more information.

World->World Testing:

This brings up a further menu enabling you to test certain aspects of the plug-in world data for errors and conflicts. Some things to keep in mind, each of these tests will test the entire game world and your plug-in. A couple of these will take some time. If you just made a small change, wait and test your plug-in in game, or wait until you have more changes to check and then run a test. These are great functions to help you debug your plug-in.

Test Models:

This will test all 3D models of all references in the world and your plug-in. If an object is missing part of it's 3D mesh or texture, this should discover that and when done testing will provide you with a list of all such errors it found.

Test Icons/Textures:

This will test all Icons and Textures for missing textures or Icons. If it finds any missing it will tell you so you can correct them.

Test All Cells:

This will test every cell in the game world, landscape and objects, checking for any possible errors. This will take a *very long* time to finish.

Test Interior Cells:

This functions in the same way, as does Test All Cells, except it only applies to interior cells. Takes quite a while.

Test Path Between Cells:

After you choose a Starting Space and a Destination Space this test will ensure that NPC's can travel along the PathGrid to get to that cell.

Output Model Size List:

I am not sure what this does, but it takes a very long time. If you do know what it does, please email me so I can place it into this manual.

PathGrids:

These tests fix various problems with PathGrids that have been created.

Update Distant LOD Data:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Update Bounds:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Update Model Texture Lists:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Recalculate Land Normals:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

World->Edit Cell Path Grid:

This brings up a box you can use to add nodes to the landscape, and link them. This creates paths for NPCs and creatures to follow.

World->Path Grid Generation:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

World->Run Havok Sim:

If you have an object selected in the Render Window that has Havok Physics, it will fall just like in game physics. This will work on NPC’s with their health set to 0.

World->Landscape Editing:

This is what you use to modify and change the landscape in the CS. Refer to the Landscape Editing Window for more information.

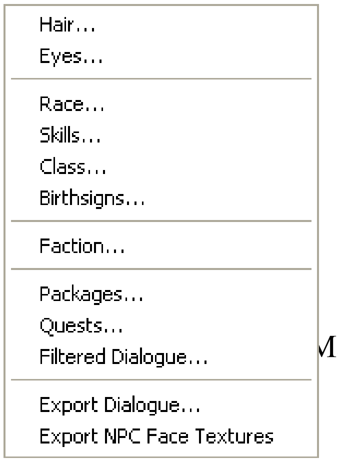
World->Heightmap Editing:

This is what you use to modify and change the heightmaps used in the game. Refer to the Heightmap Editing Window for more information.

World-> Create Local Maps:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Character Menu:



Character->Hair:

This brings up the hair window; here is where you add new hairs into the game. Refer to the Hair Window for more information.

Character->Eyes:

This works basically the same way as the hair window; which isn't very complex. Refer to the Eyes Window for more information.

Character->Race:

This brings up a window within which you can edit, create, or delete a race. This does not create, edit, or delete a body for a race; this only defines the data about a race. Refer to the Race Window for more information.

Character->Skills:

This brings up a window to modify skills, somewhat. You cannot add or delete skills. Nor can you change what actions are associated with a skill. Refer to the Skill Window for more information.

Character->Class:

This lets window lets you create you own classes that you can add into the game. Refer to the Class Window for more information.

Character->Birthsigns:

The Birthsigns Window allows you to assign spells, an image, and a description to a Birthsign for someone to choose from when creating a new PC. Refer to the Birthsign Window for more information.

Character->Faction:

This window is where you can modify existing factions or create your own. Refer to the Faction Window for more information.

Character->Packages:

This menu allows you to edit or make your own AI packages; these are the NPCs Schedules. Refer to the AI Window for more information.

Character->Quests:

This window controls all the quests in the game; here you can make your own or modify existing ones. Refer to the Quests Window for more information.

Character->Filtered Dialog:

The Filtered Dialog Window allows you to view the dialog for any NPC in game. Up the top left corner you will see a small dropdown box; this is where you choose the NPC to look at their dialog. There is more information about dialog at the Quests Window.

Character->Export Dialog:

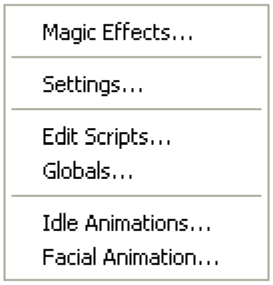
Exporting the dialog will export the entire dialog from your loaded plugins; this may take a while if you have lots of dialog in your mod and if you have a low-end machine.

Note: *This also exports the entire Oblivion.esm dialog, so be careful.*

Character->Export NPC Face Textures:

This exports Face Gen Textures for all NPCs; this will take a very long time to do if you have lots of NPCs or a low-end machine.

Gameplay Menu:



Gameplay->Magic Effects:

The Magic Effects Window allows you to change how magic appears and sounds in game when cast as a spell, the particle visual effect, and the icon showing it as active. This window does NOT change or alter spells! It changes the in game visual effects associated with a 'spell effect' that is used to create spells. Refer to the Magic Effects Window for more information.

Gameplay->Settings:

There are four tabs on this window; Gameplay, Magic, Stats, and Menus. Each tab displays similar game settings. I will not describe here what each setting does or how it relates to the game.

Gameplay->Edit Scripts:

You can add or delete a script from this menu. Refer to the Scripting Window for a description on getting started with scripting.

Gameplay->Globals:

This is a list of global variables in the game that can be referred to in scripting, some from the console, and some from the Dialog Window. Few things will cut the FPS during gameplay than adding global variables to your mod. Consider adding a new global variable to this list very, very carefully. It's up to you.

Gameplay->Idle Animations (Animation Manager)

The Animations Manager is what tells the characters in game what animations to perform. Refer to the Animation Manager Window for more information.

Gameplay->Facial Animations

This recreates all dialogue/facial animations in the game which means this will fix any dialogue animations that are missing or have been lost or replaced. My game had no facial animations until I done this.

Help Menu:

There are only two options, about tells you the version of the CS and gives you a picture (seen as the first page of this manual). The other is a link to *cs.elderscrolls.com*. This is the website to go to if you need to find out something… but now you have this manual.

The Buttons:

The buttons are merely shortcuts to commonly used functions within the CS. I will not go into depth for each function associated with one of the buttons, I will only specify what it does, or refer you to the section that describes that function.

Data: 📂

Opens the Data Window. Refer to File->Data.

Save:

Saves your plug-in. Refer to File->Save.

Preferences:

Opens the Preferences Window. Refer to File->Preferences.

Undo:

Made a mistake? Just click this button.

Redo:
Oops, you meant to do that, it wasn't actually a mistake. Redo what you just undid.

Use Grid Snap:

Toggles on the Grid Snap function. Refer to Files->Preferences.

Use Angle Snap:

Toggles on the Angle Snap function. Refer to Files->Preferences.

Heightmap:

Opens the Heightmap Editor window. Refer to the Heightmap Editing Window for more information.

Landscape Editing:

Click this button to bring up the Landscape Editing engine. Be advised, this will prevent most functions in the CS except those related to landscape editing, as this engine is a system resource hog. Refer to the Landscape Edit Window.

Edit Cell Path Grid:

Click this button to edit the Cell Path Grid. This will allow you to set down path grids.

Enable Havok:

If you have an object selected in the Render Window that has Havok Physics, it will fall and hit other objects just like in game. This will also work on NPC's with their health set to 0.

Bright Light:

Toggles brightness in the Render Window. Refer to View->Bright Light.

Sky:

Toggles the sky display in the Render Window.

Leaves:

This will display or not display the leaves on trees, take this off if you have a low-end machine. It increases performance considerably.

Quest:

This displays the quest window for editing all the quests in the game; here you can make your own or modify existing ones. Refer to the Quests Window for more information.

Filtered Dialog:

There are many functions and different drop down menus within the Dialog Window. I will not explain how to do dialog in this section. The Quests Window is where you should handle the entire of the dialog. Refer to the Quests Window for more information.

Scripts:

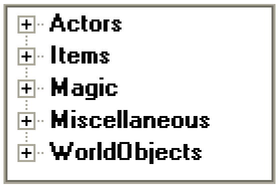
Again, I will not go into depth on the Script Window. To see how to write scripts refer to the Scripting Window.

Chapter 4 - Windows

In this chapter I will discuss all the main windows the CS uses to function (i.e. Object, Render, Cell View), plus I will go into depth on quite a few others that in the last chapter I only provided small pieces information on (i.e. Dialog, Scripting, Quests, etc.). I will provide, as much information as I can, along with any tips and tricks I have come across, but that doesn't mean this is a complete guide. There is no doubt much I do not know, or have not discovered, in the CS. There are far more savvy people out there who can tell you much more than I about any single function of the CS. I merely have tried to get it in one place, for basic instructional use by those unfamiliar with the CS, and as a reference for the rest of us.

Note: (In this note I will use the races window as an example) In almost all, if not all, the windows there is a list to the left, this has the IDs of the Races. To create a new, edit or rename a race simply R-Click on the list and press new, rename or delete.

The Object Window:



Most objects in the game that can be considered an 'item' are listed within the Objects Window, grouped in a dropdown menu, though many are not (Path nodes, scripts, Birthsigns, etc can be considered objects in some ways, and are things that have to be created or put into the game, but not from the Object Window). Some items are passive; others are active. Passive meaning you cannot 'use' the item (like a wall). Some items can be changed from active to passive in nature. Scripting could possibly be used to make a passive item operate in an active role, though that takes some original and creative thinking.

The object window contains all the objects, characters, items you can add to the game. You can also add items into the game by going to the correct place and right clicking then selecting new. The navigation in the Object Window is intimidating at first but becomes simple after a while. The object window is separated into 5 categories:

- **Actors** – All the creatures & NPC's in the game.

- **Items** – Contains all the items the player can interact with and use. (Armor, Weapons, Books, etc.)

- **Magic** – Contains the spells, potions and enchantments.

- **Miscellaneous** – Contains the sounds, effects, combat styles, and animations.

- **World Objects –** Contains the things the player can't pick up or use. (Houses, doors, furniture, beds, and activators.)

Things you should know about the Object Window:

Creating New Objects:

Don't create a new object unless you need to, use what the game provides if you can. But if you need to change an object please do create a new one. Just rename the ID of an object and when a prompt comes up asking if you wish to create a new object, click 'Yes'. If you click 'No', the old object effectively no longer exists, and all references of that object take on the new name you gave it. Most likely this will cause problems; not only that, but you can't just rename the object back to it's old name. This creates a dirty mod.

Naming New Objects:

When you name a new object ID use a unique name that no one else will likely use. Use your initials somewhere in the name to identify them as an object you created. If you use a name for an object in your mod that someone else also uses in his or her mod, and someone loads both to play, they will conflict, probably screw up their game, and maybe cause a CTD.

Deleting Objects:

Don't delete objects! It's ok to delete a reference, assuming it isn't important to a quest or the game in general (and you better be sure about it), but there is no reason to delete an object based in the `.esm`. If you want to delete an object *you* created, go right ahead. You can delete objects from an `.esp`; at worst it will screw up a plug-in.

Stat Window for Objects:

To bring up the statistics window for any object you can D-Click that object, or R-Click on it. This will bring up a menu with several options on it; New, Edit, Delete, etc. You do not get this stat window if you click on a reference, which will bring up a Stat Window.

Object Meshes and Textures:

Where are they? They are located in the .bsa files. You need to unpack them with a BSA utility.

Adding an object:

To add an object to the world simply click and drag it to the Render Window where a reference will be created of it. If you need to add several objects, highlight those you wish to add and then click and drag the group of objects. (Of course now you have to sort and place said objects in the Render Window).

Activator:

These are objects that activate something, usually by script, though some things are hard coded to operate a certain way. Often these are static like objects. Signpost, barriers, beds, etc. are all considered activators. You can attach scripts to activators. (Activators are located in WorldObjects->Activator)

Mesh:

The top button is for the mesh of the activator. The meshes are found in the .bsa file called meshes.

Looping Sound:

This is the sound played when the object is activated. You don't need to have a sound though.

Dangerous check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Quest Item check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Ammo:

The ammo window lets you edit or create new arrows in the game. (Ammo is located in Items->Ammo)

Enchanting:

Click this drop down menu to see a list of all enchantments from the Enchantments tab, choose one to add to an arrow.

Enchantment:

This is the maximum potential magicka pool an arrow can have.

Ignores Normal Weapon Resistance check box:

If checked, this weapon can hit any NPC or creature with the Resist Normal Weapons spell effect active, either as a spell or an ability.

Quest Item check box:

If this is checked then the item cannot be dropped or sold in game; the player is forced to keep it.

Damage:

This is the attack power of the arrow; be sure to make it balanced.

Speed:

This is the speed that the arrow travels through the air.

Apparatus:

These are the objects are used to create potions in game. You can attach scripts to them. (Apparatus is located in Items->Apparatus)

Type:

This drop down menu lets you select what type of apparatus it is, there is either Alembic, Calcinator, Mortar/Pestle or Retort.

Weight:

This is the weight of the object; this is how much it will take up in the PCs inventory.

Value:

This is the monetary value of the item if bought or sold. Keep in mind you will pay more than this if buying, and will get less than this if selling.

Quality:

This is how 'good' the apparatus piece of equipment is:

Novice = 10
Apprentice = 25
Journeyman 50
Expert = 75
Master = 100

Mesh:

The top button is for the mesh. The meshes are found in the .bsa file called meshes.

Icon:

The bottom button assigns the icon for the inventory. Found in the .bsa named textures.

Quest Item check box:

If this is checked then the item cannot be dropped or sold in game; the player is forced to keep it.

Armor:

You must understand how armor is rendered in the game to grasp how to modify or create new armor. When the PC or an NPC wears armor (or clothing), the armor is not placed over the body part it is assigned to, it replaces that body part. The game is not rendering a body part 'under' the armor, only the armor. What this means is you must assign parts of armor under the Biped Object section (even if only one part). This may be confusing, but if you look at a few pieces of armor already listed, it will make sense. Double clicking on an armor object opens a window to define stat values for the armor for use in the game. The easiest way to create new armor types is to simply open up a piece of armor, change the object name of it (as explained in a bullet at the beginning of this chapter), and click 'Yes' when asked if you want a new object. If you want a full suit, make sure you do this for each piece of armor (i.e. a new type of steel armor? Make sure to create a new set of gauntlets, boots, greaves, a cuirass, helmet, and shield). (Armor is located in Items->Armor)

Script:

You can attach a script to the armor here.

Weight:

This is the weight of armor. Here is a table showing, as best I can determine, the weight classifications. Armor types are listed on the left; the maximum weight that can be assigned to that armor type and still be considered of a weight class is listed across the top.

Table 4.0 Armor Type Weight Allocations

	Light Maximum Weight	Medium Maximum Weight	Heavy Minimum Weight
Boots	12	18	Above 18
Cuirass	18	27	Above 27
Gauntlets	3	4.5	Above 4.5
Greaves	9	13.5	Above 13.5
Helm	3	4.5	Above 4.5
Shield	9	13.5	Above 13.5

Heavy/Light:

In this drop down menu you select if you want the armor heavy or light.

Health:

This is the ‘hit points’ of the armor. The Armorer skill and a repair hammer are used to restore these points. If a piece of armor is reduced to zero, the armor will become unequipped and useless.

AR:

This is the value of the protection the armor gives. This number is set upon a skill level base of 30. This means that if the appropriate armor skill is 30, and the AR of a piece of armor is set to base 50, the armor has an in game current value of 50. If the skill is lower than 30, then the value of the AR will also proportionally be lower. If the skill is above 30, then the AR value will also be proportionally higher. Try to assign values that are balanced.

Value:

This is the monetary value of the item if bought or sold. Keep in mind you will pay more than this if buying, and will get less than this if selling.

Enchanting:

This drop down menu lists enchantments from the ‘Enchantment’ section. Choose one to add to your armor to make them magical.

Quest Item check box:

If this is checked then the item cannot be dropped or sold in game; the player is forced to keep it.

Playable check box:

This is whether or not the armor is playable in game; if it is not playable the PC will not be able to pick up the armor.

Hide Amulet/Rings check boxes:

If these are checked then the armor will make any amulets and rings the PC or NPC are wearing disappear; although they will still be equipped

Biped Object:

These drop down menus are used to assign the parts that this armor covers. Note that some armor covers more than one body part (ie. boots cover both the left and right foot, and also the left and right ankle). Look at other armor types if unsure what to include.

Male section:

This section contains the following buttons; I will explain each as well as I can (These apply for the female section as well):

Biped Model button:

This is the mesh seen in game when equipped on the PC or NPC.

World Model button

This is the mesh seen when the armor is dropped on the ground/

Icon image button:
The lower button is used to assign the icon for the inventory.

Books:

Books are easiest to modify or create by just renaming one and inserting your own text. They can do a couple of other specific things. A ‘book’ can actually be a scroll or a book. Scrolls can also cast a spell (one time). You can attach a script to a book (or scroll). (Books are located in Items->Book)

Weight:

This is the weight of the book.

Value:

This is the value of the book.

Teaches:

You can click this drop down menu and choose a skill from this list. The first time a copy of this book is read by a PC they will receive a one-point increase in that skill.

Mesh:

The upper button is used to assign this file. Found in the `.bsa file named meshes.`

Icon:

The lower button is used to assign this file. Found in the `.bsa file named textures.`

Scroll check box:

Click this box if the object is a scroll. Make sure to use the correct mesh.

Can't Be Taken check box:

If this box is checked then the book/scroll cannot be picked up or stolen.

Quest Item check box:

If this is checked then the item cannot be dropped or sold in game; the player is forced to keep it.

Clothing:

The Clothing Window works exactly the same as the Armor Window (detailed above) works. The difference is what body parts are covered by types of clothing. (Clothing is located in Items->Clothing)

Shirt	Upper Body.
Skirt	Lower Body.
Pants	Lower Body.
Shoes	Foot.
Robe	Upper Body, Lower Body.
Ring	LeftRing or RightRing.
Amulet	Amulet

Container:

These objects are barrels, chests, boxes, etc. Containers cannot be picked up or moved. You can remove and place items in containers. You can add a script to a container. (Containers are located in WorldObjects->Container)

Quest Item check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Respawns:

If this is checked, objects inside the container will respawn about every 4 months or so.

Weight:

The total weight of all objects within a container cannot exceed the weight value listed here. To the right is listed the encumbrance so you don't exceed this value.

Inventory List box:

Here is where you put items to be found in the container. Just open another tab on the Object Window and drag and drop an object into the list. L-Click once to highlight, and then L-Click again (not a D-Click) over the Count column to adjust the number of objects found in the container. If you enter a negative number the container will respawn stated amount of that item (which means they will always have that many, no matter how many times you remove the objects). Infinite items respawn after 24 hours.

Creatures:

The Creature Window is used to modify and create the beasties that annoy and chew on our PC during the game (Ain't they cute when they do that). You can also attach a script to a creature. If you want to modify and/or create new creatures I suggest you write down the information for that creature first. Get it the way you want it. Once satisfied, then open up this tab, R-Click to open a creature or create a new one, and fill in the data. Try to keep the balance of the game and your creature in mind. (Creatures are located in Actors->Creature)

Type:

Decide if the creature is a *Creature, Daedra, Giant, Horse, Humanoid*, or *Undead*.

Weapon & Shield:

If you check this box, and you indicate they can use the Biped animations for movement, the creature will use the best weapon and shield you include in their inventory (if you give them any).

Level:

This is the level of the creature. This is 1 by default; the CS does not compute this for you. If you forget to change this, the creature will stay level 1.

PC Level Offset check box:

If this is checked then the Creature will always be around the same level of the player.

Combat Style:

This is the Combat Style used by the Creature; you should just keep this on default.

Death Item:

This is an Item that appears on the Creature's corpse when they are dead, it will only be found there when the Creature is dead.

Movement:

Flies check box:

Check this if the creature flies… in the air of course.

Biped check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Walks check box:

Check this if the creature walks.

Swims check box:

Check this if the creature swims.

None check box:

Check this if the creature does nothing… it just 'lays' there.

Quest Item check box:

If this is checked then the creature's corpse will not disappear, it will stay there forever.

Essential check box:

If checked the creature will be tagged essential; this means that it cannot die, if you attack it in game and their health reaches 0 they will only become 'unconscious' and then get up off the ground.

Respawn check box:

If checked then the creature will respawn after they have been killed. It takes 7 in game days before the creature will respawn. (The creature will respawn at his/her original position)

No Head/Left Arm/Right Arm check boxes:

Check these if the creature doesn't have a head or left or right arms. This is only used on zombies in game. This doesn't actually remove the creature's arms or head, this only applies for the biped, but you should check this if your creature doesn't have arms or a head.

No Low Level Processing check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

No Combat In Water check box:

If this is checked then the creature will not be able to attack while in water.

Can Corpse check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

No Shadow check box:

Check this if you don't want the creature to cast a shadow.

Dialog button:

Creatures do not have any dialog by default. If you choose to add it click this button to open the Dialog Window, filtered to this creature.

AI button:

This is where you add AI Packages to the creature; you can also change their Aggression, Confidence, Energy Level, Responsibility and more. Refer to the AI Window for more information on this.

Stats Tab:

Attributes:

These are the stats for this creature. You can set these as low as 1, or as high as 255. Most creatures should have a stat value between 1-100.

Soul:

This is the value of their soul for charging a soul gem.

Skills:

The creature only has three skills, *Combat*, *Magic*, and *Stealth*. This is the skill level used for any skill from one of those categories. You can set these manually. Maximum skill level is 255.

Attack Damage:

This is the creatures attack damage; be sure not to set it to high or to low, keep the game balanced.

Attack Reach:

This is the reach of the attack from the creature.

Factions Tab:

This is where you can drag a faction from the faction window and place it here to assign the creature to a faction. Refer to the Factions Window for more information.

Inventory Tab:

This is where you can drag an object from the object window and place it here to put it in the creature's inventory.

SpellList Tab:

This is where you can drag a spell from the Spellmaking Window and place it here; the creature will cast this spell if he/she has high enough stats and attributes.

Animation Tab:

This is where you can preview animations on the NPC. Make sure you check the full check box to the right; just D-Click on the animation to view it.

Model Tab:

This is where you can assign meshes to the creature; in the list you can D-Click any of the meshes to activate or deactivate it.

Blood Tab:

This is where you choose what blood the creature has; choose a mesh and texture with the two buttons there.

Sounds Tab:

These are the sounds that the creature makes when it walks, fights etc.

Doors:

Doors are just that, doors. You can attach a script to them. When you place them remember you need a doorframe for them to fit into. (Doors are located in WorldObjects->Door)

Open:

Click on the middle button to assign a sound for when the door is opening.

Close:

Click on the bottom button to assign a sound for when the door is closing.

Quest Item check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Hidden check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Oblivion Gate check box:

Check this if the door is an Oblivion gate.

Minimal Use check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Automatic Door check box:

Check this if the door opens and doesn’t teleport the PC or NPC somewhere; this makes the door an animation.

Randomly Teleports to these Interiors/World Spaces only:

If you put interiors in here the door will teleport you to any one of these; it chooses randomly which on. Just drag and drop the interior or world space into here.

Enchanting:

The enchanting window is used to make enchantments for objects. (Enchantments are located in Magic->Enchantment)

Type:

In this drop down menu you choose what the enchantment is for; you can choose Apparel, Scroll, Staff or Weapon. (Apparel is clothing and armor)

Effects List:

To the right of the window is the effects list, R-Click here and press new to bring up the Effect Item window. Here is some information on the Effect Item Window:

Effect:

This drop down menu controls what the effect is. There are quite a few of these; some are very interesting.

Range:

This changes depending on what the Type of enchantment is, they are as follows:

Apparel = Self

Scroll = Self

Staff = Target

Weapon = Touch

Area:

This is the area that the attack affects; if this is very high and you cast a lightning bolt at an enemy it will hit enemies all around that enemy, be sure not to make this to high!

Duration:

This is how long the enchantment lasts; for Apparel it lasts forever.

Magnitude:

This is the power of the enchantment, if you have a fortify skill 'Blade' enchantment that has a magnitude of 10, it will set the PC or NPCs Blade skill +10.

ActorVal:

This drop down menu only appears when the effect is a Fortify/Drain Skill or Attribute. This controls what skill or attribute you wish to affect for the chosen effect.

Script Effect Info:

The Script Effect Window only becomes visible when the Effect for the enchantment is set to Script Effect.

Script:

This is the script you wish to be assigned to the enchantment.

Effect Name:

This is the in game name of the effect.

School:

This is what school (Skill) the effect is dependent on.

Visuals Effects:

This controls what visual effect is cast by the effect; this would usually be none.

Effect is Hostile check box:

If this box is checked this effect is classed as hostile; if you cast it on somebody you will get in trouble with the guards.

Ingredients:

These objects are able to instill a limited spell effects if eaten or used, or mixed with other ingredients to create potions. You cannot determine the amount of an effect; this is determined by the PC's Alchemy skill. You can also attach a script to an ingredient. (Ingredients are located in Items->Ingredient)

Weight:

This is the weight of the Ingredient.

Quest Item check box:

If this is checked then the item cannot be dropped or sold in game; the player is forced to keep it.

Food Item check box:

If this is checked then the PC or an NPC can eat the ingredient.

Results:

Up to four spell effects may be assigned to an ingredient, one for each drop down menu. Only the first one listed will have any effect if eaten.

Leveled Creatures:

This list acts as a reference in the game world. When the cell loads the PC's level is checked against this list. If a creature is on the list that meets the parameters as set on this list for the PC, that creature is added to the world at the reference's grid. If no creatures on the list meet the parameter, no creature appears. (Leveled Creatures are located in Actors->LeveledCreature)

Marker Script:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Creature Template:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

List box:

This is the listing of creatures. Drag and drop creatures (or an NPC) into this box. They will load top to bottom, and cannot be shifted about in the list. Each creature listed must be set to a level equal to or higher than the creature above it. You can add another leveled creature list in this box (no kidding, you can, it is considered a creature. Just give it a level, and if chosen, that list picks randomly from any creature within it, regardless of level associations).

Calculate From All Levels<=PC's Level check box:

If this box is not checked, only creatures that are equal to the level of the PC will be placed at that location. If checked, any creature equal to or lower in level to the PC will be placed at that location. (If more than one creature meets these parameters, one is chosen at random).

Chance None:

This is the percent chance that no creature will be chosen when the cell loads.

Object:

This is the spell that is selected in the list.

Level:

This is the PC's level at which you want this creature to appear. This must be equal to or higher than the creature above it. This defaults to 1, but can be adjusted manually. To do so l-click once and highlight a creature, then l-click again on the list and enter a level you wish.

Count:

This is how many spells of that type are in the barter menu.

Leveled Items:

This works similar to the leveled creature list. The leveled item list is placed within a container or on a creature/NPC. When the host object is activated, or opened, the game checks the PC's level and places an item, or items, accordingly. If no item meets the parameters, no item is placed. (Leveled Items are located in Items->LeveledItem)

Calculate From All Levels<=PC's Level check box:

If this box is not checked, only items that are equal to the level of the PC will be placed at that location. If checked, any item equal to or lower in level to the PC will be placed at that location. (If more than one item meets these parameters, all items will be placed).

Calculate For Each Item In Count:

If unchecked, each time the container (or creature/NPC) is opened, the same items will be placed that meet parameters. If this is checked, a random choice and number of items that meet the parameters will be placed.

Chance None:

This is the percent chance that no creature will be chosen when the cell loads.

List box:

This is the listing of creatures. Drag and drop items into this box. They will load top to bottom, and cannot be shifted about in the list. Each item listed must be set to a level equal to or higher than the item above it. You can add another leveled item list in this box (just like leveled creature lists above).

Object:

This is the spell that is selected in the list.

Level:

This is the PC's level at which you want this item to appear. This must be equal to or higher than the item above it. This defaults to 1, but can be manually adjusted. To do so l-click once and highlight an item, then l-click again on the list and enter a level you wish.

Count:

This is how many spells of that type are in the barter menu.

Leveled Spells:

The leveled spell list is placed on an NPC. When the player goes to the spell barter menu the game checks the PC's level and places the correct spells accordingly. If no spell meets the parameters, no spell is placed. (Leveled Spells are located in Magic->LeveledSpell)

Chance None:

This is the percent chance that the spell will be chosen when the player activates the spell barter menu.

List box:

This is the listing of spells. Drag and drop spells into this box form the Spellmaking Window. They will load top to bottom, and cannot be shifted about in the list. Each spell listed must be set to a level equal to or higher than the item above it. You can add another leveled spell list in this box (just like leveled spell lists above).

Calculate From All Levels<=PC's Level check box:

If this box is not checked, only spells that are equal to the level of the PC will be in the barter menu. If checked, any spell equal to or lower in level to the PC will be placed in the barter menu. (If more than one spell meets these parameters, all spells will be placed).

Calculate For Each Item In Count:

If unchecked, each time the barter menu is opened, the same spells will be placed that meet parameters. If this is checked, a random choice and number of spells that meet the parameters will be placed.

Use all Spells:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Object:

This is the spell that is selected in the list.

Level:

This is the PC's level at which you want this spell to appear. This must be equal to or higher than the spell above it. This defaults to 1, but can be adjusted manually. To do so L-Click once and highlight a spell, then L-Click again on the list and enter a level you wish.

Count:

This is how many spells of that type are in the barter menu.

Loading Screens:

The loading screen window allows you to create loading screens for any cells in the game. (Loading Screens are located in Miscellaneous->LoadScreen)

Displays only when these forms load list:

This lists all the cells that the load screen is displayed on. To add cells, open up the Cell View Window, then drag and drop a cell into the list.

Loading Screen Image:

This is the image displayed on the loading screen.

Loading Screen Text:

This is the text displayed at the bottom of the loading screen.

Lights:

The light does not have to be a physically seen item; it can just be a source of light. It can also be an item that can be picked up and carried. You can attach scripts to lights. (Lights are located in WorldObjects->Light)

FOV:

This is the Field of View; this controls how far the light shines.

Falloff Exponent:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Radius:

This is the radius in units the light shines on objects. Thus highlights and shadows will only be created for objects that fall within this radius.

Color:

This is a standard color palette. Choose the color you want the light to be.

Flicker Effect:

Chose one of these if you wish the light object to vary its brightness. They are as follows:

Flicker:

For a varied flickering effect such as that given off by a torch or campfire.

Flicker Slow:

The flickering effect is slowed down.

Pulse:

For a constant altering change in the light intensity.

Pulse Slow:

For a slower constant altering change in the light intensity.

This light can be carried:

Can Carry check box:

Check this box if the object can be picked up and carried. Make sure an icon is assigned.

Off By Default check box:

This box is normally grayed out. If you check the Can Carry box, this box becomes available. If this box is checked for an object that can be carried, when the PC puts the object down (drop) the item will not 'light up'. If this box is not checked, and an inventory Light object is dropped, that object will begin emitting light (if you put a candle down it will start to burn). If checked, the item must be equipped to begin emitting light.

Name:

This is the name of the light; this only becomes visible when the Can Carry box is checked.

Time:

Number of seconds the light will burn when equipped by the PC, and only the PC. Grayed out if the object can't be carried.

Negative check box:

If checked, the light does not emit light, but removes it. Example: you create a white source of light and check this box. When the light is lit, instead of a white color of light emitted in the defined radius, all white light is removed from within the defined radius.

Dynamic check box:

If checked, the game will apply lighting affects to moving objects.

Quest Item check box:

If this is checked then the item cannot be dropped or sold in game; the player is forced to keep it.

Spot Light check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Spot Shadow check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Misc. Items:

These are objects that don't necessarily fit under any other tab. You can add a script to these objects. (Misc Items are located in Items->MiscItem)

NPCs:

NPCs can be just as robust and detailed a character as the PC. Scripts can be added to an NPC. A great deal of information needs to be assigned for an NPC. For an average Joe it may not require much work, but for an NPC that is a focal point of a quest you may need to do quite a bit of thinking before you actually sit down and modify or create them in this window. I recommend you sit down and detail all information pertaining to an NPC on paper to get it the way you want it, before doing any typing on the keyboard. (NPCs are located in Actors->NPC)

Class:

Click this drop down menu and choose from any class. If you want a customized class you must first create it.

Summonable check box:

This is whether or not the NPC is summonable by a spell used by the player or another NPC.

Level:

This is the NPCs level. To the right of it is a check box saying ‘PC Level Offset’. If this is checked then the NPC will always be around the same level of the player.

Race:

Click the drop down box and choose which race your NPC will be. Unlike the PC, an NPC can be *any* race that has been created.

Female:

If this box is checked then the NPC will be female.

Combat Style:

This is the Combat Style used by the NPC; you should probably just keep this on default.

Death Item:

This is an Item that appears on the NPCs corpse when they are dead, it will only be found there when the NPC is dead.

Quest Item Checkbox:

If this is checked then the NPCs corpse will not disappear after a certain amount of time, it will stay there forever.

Essential Checkbox:

If checked the NPC will be tagged essential; this means that the NPC cannot die, if you attack this NPC in game and their health reaches 0 they will only become ‘unconscious’ and then get up off the ground.

Respawn Checkbox:

If checked then the NPC will respawn after they have been killed. It takes 7 in game days before the NPC respawns. (The NPC will respawn at his/her original position)

Can Corpse Check Checkbox:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

No Persuasion Checkbox:

If checked this will disable the persuasion menu for this NPC, this means you cannot ‘tell jokes’, bribe etc.

No Rumors Checkbox:

If checked the NPC wont display the Rumors topic in game.

No Low Level Processing Checkbox:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Dialog Button:

This brings up the dialog editor for the selected NPC. Refer to the Quest Window for more information.

AI Button:

This is where you add AI Packages to the NPC; you can also change their Aggression, Confidence, Energy Level, Responsibility and more. Refer to the AI Window for more information on this.

Stats Tab:

Auto-Calculate Stats check box:

Check this and the CS will calculate stats based upon race, class, and level. They will be then grayed out. Uncheck to modify them manually again.

Attributes:

These are the stats for this NPC. NPCs should have a stat value between 1-100. Maximum is 255.

Health:

Defaults to (STR + END/2) + (Level * END/10). You can change this manually. If you set the Health to zero, the NPC will be set as a corpse when the cell loads. Maximum Health is 65535.

Fatigue:

Defaults to (STR + END + AGL + WIL). You can change this manually. Maximum Fatigue is 65535.

***Base Spell Points**:*

Defaults to (INT * ability multiplier). You can change this manually. When these are gone the NPC will only use physical attacks. Maximum Spell Points is 65535.

Personality:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Hair:

This drop down menu lets you select the hairstyle for the NPC.

Eyes:

This drop down menu lets you select the color of the NPCs eyes.

Preview:

Full check box:

When checked this displays the full body of the NPC to the right of the window; this is useful when you want to see how the NPC looks with armor on.

Head check box:

When checked this displays the NPCs head.

Factions Tab:

This is where you can drag a faction from the faction window and place it in here to assign the NPC to this faction. Refer to the Factions Window for more information.

Inventory Tab:

This is where you can drag an object from the object window and place it in here to put it in the NPCs inventory.

SpellList Tab:

This is where you can drag a spell from the Spellmaking Window and place it here; the NPC will cast this spell if he/she has high enough stats and attributes.

Animation Tab:

This is where you can preview animations on the NPC. Make sure you check the full check box to the right; just D-Click on the animation to view it.

Face Tab:

This is where you can change the look of your NPC, you can change the their age, complexion, hair length and hair color. Here is a rundown of the sliders:

Age Slider:

This changes the age of the NPC, the older the NPC the more wrinkles he/she has.

Complexion Slider:

This makes the NPC more 'tanned' and old; not real old though.

Hair Length Slider:

This adjusts the length of the NPCs hair. If you make it to long, it might go through their armor or clothes; depends on what hairstyle you have for them.

Hair Color:

This brings up a standard color selection palette for you to pick a color of the NPCs hair.

Face Advanced:

This is a more advanced face-editing feature; this allows you to edit the face geometry. To change the geometry of the face, choose what you wish to change in the list to the right and then move the slider on the left up or down to change the geometry. This is quite powerful, have a mess around with it and see what you can come up with.

Potion:

Potions can have different effects applied to them; they can fortify skills or even drain attributes. The effects window works the same as the Spellmaking Window. (Potions are located in Magic->Potion)

Quest Item check box:

If this is checked then the item cannot be dropped or sold in game; the player is forced to keep it.

Poison yes/no:

If yes is written next to this then the potion is classified as poison; although, the only way to make it say yes is to have damage or decrease attribute etc effect at the top of the effects list.

Food Item:

If checked then the potion is classified as food; this is only checked for beer, mead etc.

Sigil Stones:

If you don't know what Sigil Stones are; they are the sphere stones you get from destroying an Oblivion Gate. Once you have them you can use them to enchant your weapons, they hold many different enchantments; the enchantments work the same as the Spellmaking Window so I will not be explaining how to add effects. (Sigil Stones are located in Items->SigilStone)

Weight:

This is the weight of the Sigil Stones, all of the stones in game weigh 1, I don't recommend going over 1; try to keep it balanced.

Value:

This is the value of the Sigil Stones; all of the stones in game are 0.

Uses:

Once a Sigil Stone has been applied to a weapon, the Sigil Stone disappears out of the PCs inventory. When you hover over the weapon that you applied the Sigil Stone to you will see how many uses it has. The uses are how many times the weapon can have its magical abilities repaired; once it expires the weapon returns to its original form.

Quest Item check box:

If this is checked then the item cannot be dropped or sold in game; the player is forced to keep it.

Spellmaking:

These objects are spells, curses, diseases, and spell like abilities that exist in the game. Spells and such are composed of spell effects. Spell effects cannot be directly used in the game to create magic, a spell is created and the spell effects are added to it to define what the spell does. You cannot add a script to a spell. To create a spell you must decide what spell effects to include in the spell (what it does), the parameters of those effects (how it does it), cost (in magicka), and the result. (Spellmaking is located in Magic->Spell)

Type:

This is what form the spell takes. This affects how it is applied to the object cast upon. The types are:

Ability:

An Ability is usually given to a race or creature from the Race or Creature window. An Ability is a *constant affect* set to the parameters you give any spell effects listed. An Ability cannot be sold, is not cast (it's always on), and does not need to be learned. It usually reflects a natural ability or knowledge inherent to a race.

Disease:

Common diseases, A PC can usually cast cure common disease spell effects.

Lesser Power:

-

At the moment I do not know what this is. If you do, please email me so I can place it into this manual.

Poison:

This doesn’t seem to be used by any spell in Oblivion, I do not know what this is. If you do, please email me so I can place it into this manual.

Power:

A power is a spell that a PC, NPC, or creature may cast once per day. They usually don’t cost magicka, but a cost can be assigned. These always succeed in being cast.

Spell:

Spells have a cost to cast, and may fail depending upon the skill level of the PC. The PC may create new spells, and NPCs can sell them.

School:

This changes depending on what spell effects you add in the list to he left.

Disallow Spell Absorb/Reflect check box:

If this box is checked then the spell cannot absorb or reflect another spell.

Script Effect Always Applies check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Area Effect Ignores LOS check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Immune to Silence check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

PC Start Spell check box:

If checked then the PC will start with this spell at the beginning of the game while in the prison.

Touch Spell Explodes w/ no Target check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Auto-Calculate check box:

Check this box and the CS will calculate the cost of a spell automatically.

Effects List:

To the right of the window there is a list; this is where you add all the spell effects for the spell. To add spell effects, R-Click in the list and press new. You can add a maximum of eight spell effects to a spell. They can be any effect, each with it's own defining parameters, within one spell. A few spell effects can be further defined.

Effect:

This is the selected effect for the spell.

Range:

Click to pick Self, Target, or Area. Self is whoever casts the spell. Target is whomever the spell is cast at, other than the one who casts the spell, as long as they are in line of sight. Touch is like well…. Touching someone.

Area:

This will be grayed out unless you choose 'Touch' or 'Target' for Range. The number entered represents radius distance in feet (i.e. a value of 5 = 5 foot radius).

Duration:

This is how long in seconds a spell effect will last. Duration of zero means instantaneously. Some spells can only be instantaneous.

Magnitude:

This is the degree to which the effect impacts the target. For some effects it impacts numerically in Health, an Attribute, Fatigue, etc. Others it represents distance, as in sEffectDetectEnchantment (number = feet). Others it is the level of a creature affected, as in sEffectCommandCreatures. These are described in the (very) small manual included originally with Oblivion under spells.

ActorVal:

This drop down menu only appears when the effect is a Fortify/Drain Skill or Attribute. This controls what skill or attribute you wish to affect for the chosen effect.

Script Effect Info:

The Script Effect Window only becomes visible when the Effect for the enchantment is set to Script Effect.

Script:

This is the script you wish to be assigned to the enchantment.

Effect Name:

This is the in game name of the effect.

School:

This is what school (Skill) the effect is dependent on.

Visuals Effects:

This controls what visual effect is cast by the effect; this would usually be none.

Effect is Hostile check box:

If this box is checked this effect is classed as hostile; if you cast it on somebody you will get in trouble with the guards.

Static:

These objects are usually things like rock, trees, walls, etc. You *cannot* attach a script to these objects. (Static Objects are located in WorldObjects->Static)

Quest Item check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Mesh:

The top button will assign a mesh/texture. There is no icon, as they cannot be put in the PC's inventory.

Subspace:

I am not exactly sure what subspaces are used for; they may be used to detect whether the player enters the area of the subspace, I'm not exactly sure. (Subspaces are located in Miscellaneous->Subspaces)

X, Y, Z:

These are the values of how big you wish the subspace to be, there are already quite a few subspaces so you probably wont have to make your own.

Weapons:

These are objects that are weapons. You can add a script to a weapon. (Weapons are located in Items->Weapon)

Type:

Click the drop down menu to choose the type of weapon.

Enchanting:

Click this drop down menu to see a list of all enchantments from the Enchantments tab, choose one to add to a weapon.

Enchantment:

This is the maximum potential magicka pool a weapon can have.

Ignores Normal Weapon Resistance check box:

If checked, this weapon can hit any NPC or creature with the Resist Normal Weapons spell effect active, either as a spell or an ability.

Health:

This is the ‘hit points’ of a weapon. Armorer skill and a Repair Object are used to restore these points. When the health reaches zero the weapon will become unequipped and useless until fixed.

Reach:

This is how far away from the PC the weapon reaches to strike something. Defaults to 1.00 (normal for a Longsword). This rating is not used for Ranged Weapons (arrows, darts, etc).

Damage:

This is the attack power of the weapon; be sure to make it balanced.

Quest Item check box:

If this is checked then the item cannot be dropped or sold in game; the player is forced to keep it.

Speed:

This is how fast the weapon is swung. Default is 1.00, thus a value of 2.00 would be twice as fast, while 0.5 would be half a fast.

The Animation Manager Window:

The Animations Manager is what tells the characters in game what animations to perform. Depending on the conditions set, whenever a person or creature reaches the conditions set for the animation. He, she or it will perform the animation accordingly.

Animation Group Section:

This is the body part that the animation effect uses.

Must return a file:

Must play an animation even if it isn’t appropriate If the animation manager can’t find an appropriate animation to play, it will play the next one which may not be a good idea in some cases.

Conditions:

This is what must be met before the animation will play.

Go to the Beginning

The Heightmap Window:

The Heightmap Window and associated functions will be included in a future update.

The Weather Window:

The Weather Window and associated functions will be included in a future update.

The Climates Window:

The Climates Window and associated functions will be included in a future update.

The Regions Window:

The Regions Window and associated functions will be included in a future update.

The World Space Window:

The world space window is where you can make your own town or land. As you open the World Space window you will notice the list to the left that contains all the towns, oblivion gates and even Tamriel itself. You can edit these world spaces or make your own by R-Clicking in the list and pressing new. But before you can make your own you should know what things do on this window.

Name:

This is the name of the world space.

Parent World Space:

This drop down box determines what the parent world space is to the selected world space, if you are creating a new town in Tamriel this would be set to Tamriel or a custom land.

Music:

This is what kind of music is played in the World Space. Default is exploring music; the music playing while you are walking around the Tamriel World Space, Dungeon is the music played when the player is in a dungeon and public is the music played in a town.

Sharable Data:

Climate:

This is the climate for the selected World Space. Refer to the Climate Window for more information on Climates.

Water:

This is what sort of water is in the World Space, if it is an Oblivion Realm it should be set to a lava setting.

Map Data:

I do not know anything about the map settings for Oblivion; I will try and add this as soon as possible, if you know what these settings do please email me.

Roads:

I do not know anything about these road settings; I will try and add this as soon as possible, if you know what these settings do please email me.

Cant Fast Travel From Here check box:

If checked this disables fast travel from this World Space. This is mostly used for interiors such as dungeons or houses.

Can't Wait check box:

If this is checked the player cannot wait in this World Space.

No LOD Water check box:

If checked this doesn't display the water outside of the current cell.

Oblivion World Space check box:

Check this if your World Space is an Oblivion Plane.

Small World check box:

This determines if the World Space is a town or not. If it's an island, it is not a small world. If it's a town; it is a Small World.

Output Cell Ref Counts:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

The Render Window:

The Render Window shows the game world while modifying, adding, deleting, or just plain looking at it. Moving about the 'scene' in the window can take some getting used to, learn to use the keys on your keyboard with the mouse to move about and rotate what you are viewing. The main function of the Render Window is to view the world as seen in the game, add/move/delete objects in the world (references), and/or to modify the landscape. There is not a whole lot to say about the Render Window, but here are some tips to keep in mind when working in it:

When viewing:

The Render Window is for viewing cells and references placed in it, use the Preview Window to view objects from the Object Window.

Movement may take some getting used to. Check out the Shortkeys to help you understand movement about the Render Window.

Use Bright Light if the screen is dark, but don't forget to add lighting if you do. I have a tendency to use brightness and forget that the cell I was working on was still dark in game.

When landscaping, try doing it in Wireframe mode. I found it easier on my eyes once I got used to it, and my system rendered changes quicker. I don't suggest vertex painting or texturing in wireframe mode (though you can).

Don't forget that if your movement about the window seems slow, you can adjust the speed or amount of movement under Preferences.

You can place an object so that only part of it is visible (sinking a boulder into the ground so that only a part is visible creates a small rock). This is fine and useful. However, the game will continue to render the entire boulder, even the part that will never be seen. Keep this in mind as you build. Just because it doesn't seem like much is in a cell, unseen polygons could be causing a huge drop in FPS.

Objects:

Selected objects appear to have a 'box' of red and green lines around it.

Not sure which way is North? Select an object and hit 't'. You will move to a top down view over that object and the Render Window will orient so that north is now 'up' in the window. When in an interior cell always place a Northmarker (suggest you make it your first object placed). This will help keep you oriented and the game wants it.

Northmarkers always point north. When you place one, place it so that it points the direction you *want* to be north, not what you *think* is north.

When you place a reference (of an object) you can make it bigger/smaller, or rotate and move it about, but do not changes the *stats* of the object. If you do, you are changing the stats of that reference's object. This changes EVERY reference of that object in the game. If you need to change the stats, make a new object first in the Object Window (by renaming it), and use that new object. Refer to The Object Window.

If you are having trouble placing an object exactly where you wanted it, try using Angle Snap and/or Grid Snap for big statics. For smaller items you should keep these off.

If placing several references of the same object in a window, it may be easier to place a reference and then duplicate it. You can then place the duplicates (which stack upon the original reference) where you want.

When you place your first reference in an interior, it may not be visible in the Render Window (an oddity of the Render Window). To find it, just go to the Cell View Window and double click the reference you just added. This will select the reference and you will be centered on it in the Render Window. D-click it to bring up the reference stats window. Change the 'x', 'y', and 'z' locations to 0, 0, 0.

To move a reference L-Click while you hold down the 'x' or 'y' keys when making small movements. R-click and hold, then move your mouse to rotate the reference. When doing this, or if just moving it without holding these keys, the reference will not move up and down, only horizontally. To move on the z-axis (up and down) hold down the 'z' key.

To resize a reference, hold down the 's' key, click on it, and drag up or down with the mouse, or you can manually do this from the reference's stat window.

Even though many objects are intended for use in an exterior or interior cell, almost any object can be placed in either, regardless of the original intent. Nice huh.

If you plan on adding a script to an object, *make a new object* and add the script to it. This will help keep your plug-in simple to work with, and compatible with other mods.

Building:

When building interior cells and putting buildings together in exteriors, use Angle Snap and Grid Snap when placing a reference. Make adjustments for the amount of 'snap' that work for you. This will make building much easier and quicker than trying to line objects up seamless manually, especially when you first learn to do this.

When building, remember the building blocks (walls, corners, etc) are references of objects too. Treat them as such.

The Preview Window:

The Preview Window by default is not active, the Render Window is. To open it simply click on the View Menu and then Preview Window. This window is used to view objects. It is not for landscape, references of objects, or to view anything in the game 'world' If an object has animation associated with it the object viewed will be shown animated (usually). If it has sounds, you'll hear them. Viewing can be awkward, if you have this window small you may not be able to view or find an object selected. Simply expand the window, click a different object (so it refreshes), and use the arrow keys (or page up and page down) to find an object. The arrow keys will move an object by rotating it within the window to get it to a position so that you can view it. This can be tricky for some objects. The functions listed below the grayed out Current Cell Only under the View Menu have no function in the Preview Window. This window has little other use or function.

The Cell View Window:

The Cell View Window is comprised of two parts: The left side that lists cells and the right side that lists contents of those cells.

Single click once in the Cell List (left side) on a cell to see all references (of objects) within it listed in the Reference List (right side). D-clicking a cell in the Cell List will load the cell in the Render Window along with all references in the Reference List.

A single click will highlight a reference in the Reference List (right side). A D-Click will load the cell that reference belongs to in the Render Window, and center your viewpoint upon that selected reference.

You can click upon the top of any column to change the listing under it to be listed alphabetically, or numerically, ascending or descending.

D-Clicking upon a reference in the Reference List will not bring up that reference's Stat Window, merely load the cell and center you upon that reference. You can then d-click the reference in the Render Window and bring up the Reference Stat Window.

The Hair Window:

This brings up a window, which lets you edit, create, or delete a hair in the game. This does not create or edit a hair mesh (3D Model) for a race. Here I will explain what everything does:

Name:

This is the name of the hair. (Tonsure, Topknot etc.)

Playable Check box:

This makes the hair playable. Some of the hairs that came with the game are not playable but you can easily make them so like this. If you do, do not bother releasing it to the public, there are already lots of these!

Fixed Color Check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Can Be Male/Female Check box:

These control if a male or female can use the hair in game.

The Eyes Window:

The eyes window lets you create, edit or delete eyes for in the game. Here you cannot create eye textures.

Playable Check box:

This makes the eyes playable. Some of the eyes that came with the game are not playable but you can easily make them so like this. If you do, do not bother releasing it to the public, there are already lots of these!

The Race Window:

This brings up a window within which you can edit, create, or delete a race. This does not create, edit, or delete a body for a race; this only defines the data about a race. Let's look at each box of data.

Note: *Do not make changes to the provided races without some forethought. A race should be treated just like an object, you change one thing about a race in this window and you change every reference of that race in the game. This could have unforeseen consequences.*

Editor ID (To the left):

This list displays the ID for a race. The game treats a race as an object, so the object naming conventions still apply. The ID name does not show up in the game, it is the name by which the game identifies a race.

General Data Tab:

Base Attributes:

These are the starting attributes of Strength, Intelligence, Willpower, Agility, Speed, Endurance, Personality, and Luck. You must assign a value to both males and females (male on the left, female on the right). You can assign a starting value of up to 255 for a race. The ‘normal’ starting value is 30-40. I suggest a starting total of all attributes to equal 310, as to keep the game balanced.

Skill Bonus:

These are the skills a race receives a bonus to in the game. There is seven drop down menus, with a small box to the right of each. You can assign up to seven skills for a race. Click on a box to see all the skills. Pick one and it appears in the box. Now choose a number for the bonus for that skill in the box immediately to the right. This can be any number up to ten digits, but will ignore the number if it goes above ten digits (not sure how the game would handle a big number).

Specials:

Here you can add or delete spells, powers, and abilities to a race, or change the ones they have. To add, open the Object Window (do not close the Race Window). Click on Magic->Spells, then choose a spell from the list or make a new magical spell. To add a spell to a race click and drag the spell from the object window to the Specials box in the race window. It should then appear in the box. You can highlight more than one spell and drag all as a group. To edit a spell, you do not click on the magical abilities box in the Race Window; this only displays abilities for that race. Instead edit the spell itself in the Spellmaking Window. To delete a spell ability from a race, merely highlight that spell in the spell abilities box, and then hit your ‘delete’ key.

Body Data Tab:

Male Bodies/Female Bodies:

These sections of the window control the height, weight and textures of the race. (Their bodies)

Eye Colors:

This is where you can add eyes for the race; to add eyes, simply open the Eyes Window (Keep the Race Window open) and drag the eyes from the eyes window to the Eye Colors box in the race window.

Hair Styles:

This is where you can add hairs for the race; to add the hairs, simply open the Hairs Window (Keep the Race Window open) and drag the hair from the Hair Window to the Hair Styles box in the race window.

Face Data Tab:

The face data tab contains all the 'face data' for the race. It contains all the races face textures and meshes. The left row of buttons is meshes and the right row is textures.

FaceGen Data Tab:

In this section you can change how the default races face looks; I am pretty sure it is used for something else as well; if you know what it does, please email me so I can place it into this manual.

Text Data Tab:

Here is where you can change the in game name of the race and the description (Which you see right at the beginning of the game when you are choosing your race). There is also a check box here; Playable, check this if you want the race playable.

Reactions Tab:

This tab has the races disposition towards another race. To add dispositions here, R-Click on the grid, and press new. Then to the right of the widow select the race you want and the amount of disposition.

The Skill Window:

This brings up quite a big window that shows a list to the left showing all the skills. In the center is what actions they affect in the game, a numerical value associated with an action to increase that skill, and an attribute associated with that skill. The only changes you can make to skills are the attributes associated with that skill, and the value to determine the increase of a skill.

Governing Attribute:

This box shows the currently selected attribute associated with a skill. You can change this.

Actions:

You cannot add or delete these, however to the right of each action is a box showing the value added when using this action to increase this skill. In general terms, a value of 1.00 can be thought of as 'normal'. Increase this number and you progress faster in that skill when using that action. Lower this number, and you progress slower. I do not suggest you make any changes to these values without some caution and much needed prior thinking. Most of the skills are pretty well balanced as is (hey, I said most, not all. I know a few could use some minor tweaking… that's up to you).

Specialization:

This is what the skill specializes in, either combat, stealth or magic. I do not recommend changing these.

Skill Perks Text:

This is where you can change what the game says when the PC reaches the required skill level for the perks. There is really no point in changing this.

The Class Window:

This window is used to create all classes within the game. All NPCs and the PC must have a class assigned to them, even if they are the only one with that class.

Editor ID (To the left):

This is the unique name ID of the class used by the game.

Full Name:

This is the name of the class that is used in game and seen by the PC.

Primary Attributes:

These are the two Attributes considered to be primary for the class by the game.

Specialization:

The class specializes in *Combat*, *Magic*, or *Stealth* skills.

Major Skills:

Click each drop down menu and choose a skill that will be a major skill for the class.

Playable Check box:

Check this box if you want the class available at the start of the game for the PC.

Description:

Enter a description of the class in this text box. This will appear in game.

Auto-Calc Buys/Sells:

On the NPC window, if the auto-calc box is checked, the NPC will Buy/Sell according to boxes checked in this section for their class.

Auto-Calc Other:

As per Auto-calc Buys/Sells, but for ***Training***, ***Spellmaking***, ***Enchanting***, and ***Repair***.

The Birthsign Window:

This window allows you to modify Birthsigns. Birthsigns give spells, powers, and abilities to a PC. If you wish to modify or create new image files for a birthsign, you should be able to use almost any decent graphics program. Most commonly used and/or mentioned in forums I've seen are Paint Shop Pro and Adobe Photoshop (and no, you cannot just download these free somewhere, although GIMP is for free, check Chapter 6 for links.).

Editor ID:

This is the name ID, but it is not displayed in game.

Full Name:

This is the name by which the birthsign is known and will be displayed in game.

'Constellation Image' button:

Click on it and you can alter the picture file associated with the birthsign. This is the picture shown in the Birthsign Menu where you choose your birthsign.

Spells:

Here you may drag and drop any spell from the Spellmaking tab in the Object Window to associate it with a birthsign. Highlight a spell and push the 'delete' key to remove a spell.

Description:

Here you may enter text to describe a birthsign.

The Landscaping Edit Window:

Editing of landscape can be a major undertaking. It also will use most of your system resources to do (it shuts off many functions within the CS also). Once you've gotten the hang of it, it usually takes no time at all to create or modify minor areas, something within one cell. If changing the landscape over multiple cells it might take a little longer (or maybe a lot). I have also noticed that making changes to the landscape increases the size of your plug-in quite a bit. Keep in mind that there are a lot of tutorials out there on the Internet; I'm not going to tell you how to create anything specific, just on how to accomplish changes with the Landscape Editor. First, some general information to know about the landscape within Oblivion.

> First and foremost, landscaping will add a lot of size to your mod. Creating one single new cell listed prior as wilderness will add almost 80kb to your plug-in. Start adding textures, plants, trees, actors, and other objects, and your plug-in's size will grow phenomenally. It's not the amount of change to the landscape done; it's the fact that change is present.
>
> A cell is composed of 8192 x 8192 "units". Each unit is 0.56 inches or 1.42 cm. Thus a cell is 4620 inches (385 feet) long and wide, or 11,633 cm (116.33 meters) long and wide.
>
> In wireframe mode the little triangles you see are the polygons that make up the landscape mesh. Each side of a polygon is referred to as a vertex. Each vertex is 128 units.
>
> You cannot modify the landscape horizontally, only vertically (up and down). This vertical plane is referred to as the z-axis.
>
> Water is set to a coordinate of zero (0) on the z-axis, you cannot change this fact (I know it sucks, just learn to deal with it). You cannot create waterfalls or place water above this. There are waterfall objects you can use with cells, but you cannot start water at one level and have it 'flow' to another lower level, period.
>
> The lowest point in the landscape is -2000 for a default cell. You can lower this quite a bit but not sure what the lowest point possible is. Similarly, not sure what the highest possible point is.

Buffer Cells:

The CS will by default display or create two buffer cells around any cell you are working on in the Render Window. If you zoom outward so that multiple cells are visible you can see this (push the 'b' key to toggle cell borders). If you uncheck the **Allow Render Window Cell Loads** box in the Preferences window the CS will load any cell you click on in the Cell View Window, and render those around it, but you will only be able to

---

modify the cell you clicked on, nor will you be able to scroll to adjacent cells. You will have to click on them in the Cell View Window to load them. If you click on a cell or scroll to a cell that does not have a cell adjacent to it, the CS will create cells named 'Wilderness' automatically.

Movement within the Render Window:

You can use the Mouse to move about within the Render Window, or you can use the arrow keys for quick movement. Remember you can adjust the amount of the movement values by adjusting them within the Preferences window.

Remember to assign a Region:

If you create new landscaped cells you must create and/or assign the cells to a Region. You most likely will also have to modify or add Weather, sleep options, and Sounds.

Mess with the Landscape Editor in a wilderness cell for a while before you decide to do some serious work with it if you haven't used it before (just don't save it). This will give you some experience and save some frustration when working on your plug-in. Let's look at the options within the Landscape Editor:

Edit Radius:

This number is the size of the radius that modifies the landscape. This is displayed as a red circle in the Render window. For large modifications use a number as big as 15 (largest possible) to produce deep bowl like depressions, or large rounded hilltops. Use a number as small as 1 for single simple modifications. Small numbers produce a more conical effect when the landscape is stretched up or down.

Edit Falloff %:

This number is how sheer or steep the sides are of landscape you modify. The lower the number the less steep of the edges of the land you modify.

Flatten Vertices:

Checking this box causes all land you click on to flatten to the same height within the Edit Radius. If you click and drag with this box checked you can effectively flatten mountains or level a large area of land quickly.

Soften Vertices:

Checking this box causes all land you click on to become less steep. With each click all points of each vertex will adjust to a common average level on the z-axis, or simply put the landscape will attempt to become more flat. But this is a slower

---

adjustment than Flatten Vertices. You can click and drag the mouse to soften a large area just like with Flatten Vertices.

Show Edit Radius:

Checking this box shows displays a red circle showing the landscape vertices you are able to modify. It's on by default.

Texture:

Textures are what you use to 'paint' the landscape. It is not entirely the color of the land, but also what exists on that vertex; grass, mud, ash, swamp scum, etc (though since this does add color you can think of it as 'coloring' the landscape). You can only paint one 'square' at a time. To paint a larger area, R-Click and drag the mouse over a larger area, then release. This paints a large area, however, if you click on undo only one square will be undone. That's a lot of potential undo clicking. It's usually safer to paint single squares or smaller areas. The landscape editor radius has no bearing on painting textures.

Vertex Color:

This allows you to paint colors on the landscape, usually to add shadows and/or highlight (but usually for shadows). You must check the box 'Edit Colors' to paint with vertex colors; otherwise you will be modifying the landscape height. You can paint with two colors at once; L-Click on one color and R-Click on another color. You can now paint a different color with each button. There are two Select Color buttons, one for the left and right mouse buttons. To the left of this button are the R, G, and B boxes; you can define a color by entering numerical values if you know them. If you click the Add Color button it brings up a standard color selection palette for you to pick a color. Above the button is a display box to show you the color chosen. At the bottom of this section are sixteen boxes to store a color in for repeated use so you can switch colors. Just click on one of these and store that color in the respective left or right mouse button.

Turn Off Landscape Editing:

This button will turn off the landscape-editing engine.

The Faction Window:

This window can be confusing at first, but doesn't take long to learn. A faction can be a guild, a clan, a tribe, a culture, or a religion. You can even make a faction that isn't a faction (of sorts). Use your imagination. I personally feel this to be a very powerful aspect of the game that isn't used to it's full potential by most modders out there (excepting a very few, and bless them for it). Keep in mind this window only lists the defining data for factions; the promotion, inclusion, or exclusion, of the PC is done with Dialog and/or Scripting.

Name:

This is the name by which the faction is known in game.

Hidden From PC Check box:

If checked, the faction will not be displayed or known to the PC in game (not by the game at least, you can have a topic of the same name though).

Rank Data:

Ranks must appear in the columned box in ascending order. This means the topmost listed rank is at the bottom of the list.

***Rank name*:**

The name of each rank can be changed or entered by L-Clicking on a rack so that it is highlighted, typing the female or male rank names in the boxes to the right.

Interfaction Relations:

This is where you list all other factions that you want a reaction bonus/penalty to anyone of that faction when they interact with the PC. The left column shows a faction, and the right column shows the reaction bonus/penalty. To change the faction name L-Click to highlight the name, and then go to the drop down menu to the right and click on it. Now choose the faction you want in that spot. Enter the reaction value (either positive or negative) in the box below the drop menu. If you wish to add or delete a faction, R-Click over a faction name or a blank row in the first column, choose New or Delete. If you pick new, the current faction Name will appear, ignore this and just choose a name from the drop down menu to the right, and then enter a reaction value.

The AI Window:

The AI window controls quite a bit of the NPC, it controls their AI Packages (Schedules), aggression etc. I will give you a rundown of the window:

AI Button:

This window displays the AI screen for that creature or NPC. It contains a number of flags, boxes and tabs. Along the top are four boxes that are used to set NPC character behaviours.

Aggression:

This value controls whether or not a NPC enters combat. If the NPC's disposition toward another actor (or PC) falls below this level, it will initiate combat. So the higher this value is set the more likely the NPC is to fight.

A setting of 5 or below means it will not initiate combat under any circumstance, but will enter combat if directly attacked or to protect a member of the same faction.

A setting of zero means it will refuse combat even if attacked or a fellow faction member is attacked.

Confidence:

This as a measure of how likely the NPC is to attack or flee in combat. It is a threshold for a hidden calculation. A setting of 100 guarantees the actor will never flee. A setting of 0 means he will always flee.

Energy Level:

This value determines how often the NPC will move to a new location when executing Wander packages. The higher the value the more often he will move.

Responsibility:

This value determines how willing the actor is to commit a crime.
Settings below 30 means the actor will commit a crime (i.e. steal food if given an eat package). All fences must have a value below 30 to accept stolen goods.

A setting of 100 means the actor can report crimes directly, adding bounty to the player without a guard being present. The actor cannot make the actual arrest though.

AI Package List:

This is the list of packages this actor will run. See below for more information.

Packages:

Packages are bundles of AI instructions with conditions for when and how to execute them. They are placed on an actor by dragging them to the AI Package

List. When an actor needs to pick a new package, the list is examined from the top. The first package that is valid, based on its schedule and conditions, is selected and applied to the actor.

This is very important.

Every hour unless we force an evaluation, the actor's package list is examined from the top to the bottom. The first package that has all its conditions (including a schedule) met, is executed regardless of the other packages. If you place an open ended wander package at the top of the list, the actor will keep using it.

Package Type:

Sets the basic behaviour type for the package. Each type enables and disables appropriate fields in the tabs. The behaviour type determines the animations that the character performs.

Flags:

A set of flags that modify the behaviour of the package.

Door Flags:

Any teleport doors in the cell that are owned by the actor will be locked or unlocked depending on how these flags are set. (eg. This is used to get Merchants to lock their shops at the end of a working day)

Schedule Tab:

Sets the time at which the package can run. The schedule cannot be in less than one hour blocks.

Conditions Tab:

A full set of conditions that must be true for the package to be selected. Only applies if the schedule is valid. "Run on Target" is only valid if the package has a defined Target.

Location Tab:

Defines the location for the package to occur. It is generally recommended that you only use the **Near Reference** option. If no location is selected then it defaults to the **Near Current Location** setting.

- Near Reference: Select the reference from the render window or from the lists.
- In Cell: Not recommended. It can produce unexpected behaviour.

For example if you set an *Eat package* for a Tavern Cell, the actor can eat anywhere in that cell, which may not cause any odd problems. However, setting a *sleep package* to the same Tavern will see the Actor try to sleep in the first bed they can. Most bars have beds behind the bar for the *innkeeper* to sleep on. The actor may occupy this bed, forcing the Innkeeper to leave the premises to find a bed in other cell.

Near Current Location:

Centres the package's location on where the NPC is when the package starts.

Near Editor Location:

Centres the package's location on the default location of the NPC. (i.e. where you place them in the CS)

Radius:

The distance from the package's centre that is considered valid.

Target Tab:

Some packages types have a target (reference) that is independent of the location. This takes the actor to a precise point.

Specific Reference:

Select the reference from the render window or the lists.

Any Object:

Select a particular object ID or a class of objects.

Value:

The number of these items that the package will interact with - if it is a Generic Target. It is the distance to the target - if it is a Specific Target.

The Package types available are all context sensitive. Each has its own rules for what location means and what target means. The following are instructions for the packages.

Accompany Package:

- **Target:** Who or what to accompany.

A target is required for an accompany package. The accompanying actor will always move to within **Radius** of its **Target** and then head where ever its Target is heading. You can limit this by time. A Duration of 0 means this package will not end until a package of higher precedence needs to start.

- Location : Not applicable...

Ambush Package:

Note: *The name Ambush can be misleading in that it is not an attack package. It has much in common with the find package.*

- Location: Actor moves to the location and waits for the target to enter the radius

- Target: Actor finds this target once he has entered the **radius** on the **Location Tab**.

The actor's action depends on other game settings.
The actor travels to the **specified target** and "activates" it.

The meaning of activate depends on the target -

Target	Meaning
NPC	Talk to him, or fight if you don't like him.
Creature	Attack it.
Object	Pick it up.
Container	Open it.
Chair	Sit in it.
Door	Open it. Go through for load doors.

Cast Magic Package:

It can be a specific spell which they will cast even if it is not in their spell list. If you pick an object type to cast, it will only use their spell list.

Location:

Target of the **spell** (only valid if casting a touch or target spell). The location is where you want the spell cast. This means if the location is a reference and the spell to cast is a touch or target then that is the intended target.

- Target: Spell to cast.

NOTE: The **Target** of the **package** is the spell you want to cast - not who or what you want it cast it on.

Eat Package:

In order to eat the actor must have an ingredient that restores fatigue in his container. If he/she does not have the food item, they will attempt to gain the object by buying, killing, stealing or finding it. No actor with a responsibility above 30 will steal or kill for food unless it is a creature. If they can not find food they can use, they will go into a wander state (acts just like wander behaviour) and search for food again periodically. Remember there is no physical reason for an Actor to eat. He will not die. This is used to add life-like behaviour to actors, not as an essential part of the game mechanics.

- Location: Where to sit, which can be as specific as a particular chair in the tavern or as general as the town tavern cell. In general, NPCs will attempt to find somewhere to sit while eating.

- Target: If the eat package has a target, the actor must have or find that target to eat it. This must be an ingredient.

Escort Package:

The NPC acts as the lead in moving to a location. If the follower falls behind by a specified distance, the escorting actor will stop and wait, doing a "come along" animation or travelling to the actor to get back within range. The wait distance is determined by two game settings, one for indoors and one for outdoors. Also, the wait distance is set on the Target tab. It is recommended that the wait distance be a value greater than 400.

(If the Player is being escorted, he does not get a follow package forced onto him.)

- Target: Who/what to escort. An escort package will add a follow package onto the target if the target is an actor.

- Location: is the destination of the escort package. A location is required to make an escort package.

Find Package:

This is similar to the Ambush package. It differs because Find expects the target to be at the location, while Ambush waits for the target.

The Actor travels to within the specified **Radius** of the **Location,** then finds the target and "activates" it. The meaning of activate depends on the target.

Target	Meaning
NPC	Talk to him, or fight if you don't like him.
Creature	Attack it.
Object	Pick it up.
Container	Open it.
Chair	Sit in it.
Door	Open it. Go through for load doors.

- Location: Optional (if none specified, default is Current Location).

When the actor reaches the location, it does a search (within radius = fAIAcquireObjectDistance) and builds a list of targets up to the specified Count. It then activates each of them.

If it runs out of targets before it has reached Count, it will wander within Radius of the Location for the duration of the package, periodically rechecking the area for suitable targets.

- Target: The NPC will Travel towards the target until it gets within Target Radius, then it activates the target. The count setting determines how many of these the actor will attempt to find.

Flee Package:

This will cause an actor to run **to** a location or **from** a target. If you don't put a target on the package then the actor will flee to a location and remain there running the cower animation. If you put a target on it, then the actor will run away from a target keeping the distance that is specified in the target data. If he is the right distance from the target he will stand and cower. This package will not end until a new package evaluates.

- **Location:** Where to flee to. Ignored if target is specified.

- Target: Who or what to flee from. **Distance** is the distance the actor will try to stay away from his flee target.

Follow Package:

This sets up the opposite relationship to escort (ie NPC follows target).

- Target: Who/what to follow. A target is **required** for a follow package.

- Location: The destination of the follow is not required, but can be applied. A follow can end by reaching the **destination** or if the **duration** runs out. A Duration of 0 means this package will not end until **a package of higher precedence** needs to start.

Sleep Package:

- Target: Not used.

- Location: Where to sleep. It can be as specific as the third bed from the left in the mages guild or as general as Tom's house (bed must be a persistent reference). If it is a general location then the actor will sleep in any bed in that location that is not owned by another NPC (unless they have a low responsibility) and is not occupied.

- If they can not find a bed they will go into a wander state, periodically checking for a bed.

Travel Package:

- Target: Not used.

- Location: Go to specified location. If the quest conditions force the actor to use this package again, he will just stand there.

If a furniture object (such as a bed or chair) is selected then the actor will lay in the bed or sit in the chair. The **Duration** and **Flags** affect the outcome of this package.

--- Duration = 0: No Flag. Travel until you reach location, but could be overtaken by a higher priority package before getting there.

--- Duration = 0: Must Complete. Travel until you reach location. Will not give way to any other package until location is reached.

--- Duration = 0: Must Reach Location. Same as **Must Complete**, but will evaluate immediately, when at location, instead of waiting for normal time interval.

--- Duration > 0: No Flag: Travel until you reach location, but could be overtaken by a higher priority package before getting there.

If NPC reaches the location before end of duration, he will stay there (but could give way to a higher priority package and leave again).

If they don't reach location before end of duration, they will re-evaluate and could change to new package.

--- Duration > 0: Must Complete Flagged. Travel until you reach location. Will not re-evaluate to any other package until location is reached **AND** duration expires.

--- Duration > 0: Must Reach Location. Travel until you reach location. Will not re-evaluate to any other package until location is reached. If the NPC reaches the location before end of duration, it may re-evaluate to higher priority package if applicable.

UseItemAt Package:

- Target: The item to be used. It can be in the actor's inventory or in the world near that location.

- Location: The location to travel to, before attempting to use the target object.

If the idle animation is set up properly, the actor will play the idle associated with the item. (See Later for an example in this quest)

Wander Package:

- Target: Not Used.

- Location: Go to specified location and wander near it, within specified radius. Radius must be greater than zero. Recommended radius is greater than 100.

Actors in wander packages are also trying to find people to talk to within their social radius. If you **do not want** an actor to engage in dialogue in a wander they must have **skipfallout behavior** checked.

Also the energy level of an actor determines how often they move in a wander. If they find a chair in their wander radius they will likely sit down for a while.

Again **Duration** can affect behavior .

--- Duration = 0: No Flag
Wander until something higher evaluates.

--- Duration = 0: Must Complete. Meaningless.

--- Duration = 0: Must Reach Location. Will not re-evaluate to any other package until location is reached. When at location, wander until something higher evaluates.

--- Duration > 0: No Flag.. Wander at location until duration expires, unless something higher priority evaluates earlier.

---- Duration > 0: Must Complete. Wander at location until duration expires. Will not re-evaluate to any other package until location is reached AND duration expires.

---- Duration > 0: Must Reach Location. Will not re-evaluate to any other package until location is reached. When at location, wander for remaining duration unless something higher evaluates first.

Additional Notes: For the NPC to not run into walls, you must first create a **"priority"** path. To do this click on the path mode button then while holding both **CTRL and ALT**, lay out a path that you want the NPC to follow.

The Quest Window/Dialogue Window:

While it may seem confusing when first approached, in concept it really isn't that difficult. But it can be very, very easy to screw up a quest or how actors function, and not even realize it, if you don't pay good attention to what you do within the Quest Window. I recommend having at least a basic idea how scripting works before you dive into Quests. Quests works pretty much the same way, but are 'viewed' entirely different, and provided and accessed in a simple to use window (instead of writing everything out, you just click the drop down menus to choose IDs, functions, and conditions, etc). Once you learn how to do one of them (quests or scripting), the other becomes easier to understand. Some people find scripting easier to understand, others think dialog is easy, but struggle with scripts. I myself struggled horribly for the longest time trying to understand dialog and quests, but once it clicked, I had no trouble with it. Many have prefixed names that refer to the main storylines they are part of.

MQ - Main Quest
Arena - Arena Quest
MG - Mages guild
Etc.

MS refers to *Miscellaneous Quests.*

Creating a new Quest:

Once you open the quest window you will see a big list of all the quests to the left, R-Click in here and press new. A pop up will appear, here is where you enter the ID of your quest. Once you enter an ID press ok.

Quest Data Tab:

Quest Name:

This is the name of the quest as it will appear in the player's journal. In effect, this is the chapter heading for your part of the players story.

Priority (value 0-100)
I've read a lot of confusing and conflicting advice about the *Priority Setting* you should select. When you talk to an NPC, or they talk to each other, their *dialogue* is assigned by the priority. The dialogue for higher priority quests will take precedence over the dialogue of lower priority quests.

This is most obvious in the greetings you receive as you progress through the game.

The table below shows the conventions used in Oblivion:

Priority	Type
1	Tutorials
5	Generic Activities quests
10	Non-quest Dialogue
20	House & Horse quests
40	Training & Vampire quests
50	Miscellaneous quests
60	Guild quests
85	Main story quest
90	Crime & Character Gen quests

This is not a set of hard and fast rules but a convention. You can break it and set your priority to any level up to **100**. But, it seems **sensible** to **stick to the convention** unless you have a very good reason to break it.

Start Game Enabled check box:

If this box is checked, the quest is active at the beginning of the game. If you scan through the list of quests you will find that a very large number of quests including this one are *Start Game Enabled.*

In other words these quests start as soon as you begin the game and find yourself in that prison cell. They run quietly away in the background until an event triggers them to show themselves.

If we add a mod that is *Start Game Enabled*, it will be up and running as soon as a user loads it. We need to be aware of the consequences of this in terms of scripting. If we choose not to check this box, the quest will remain inactive until we use the correct Script command (**StartQuest** or **StartStage**) to set it up and running.

Allow Repeated Conversation Topics check box:

By default each conversation uses a convention where any selected topic is considered completed once you have selected it. It is greyed out and not visited again once we have selected it. If you click on **GREY FOX** in the topic list of a NPC, he will respond, and then the *Grey Fox* topic is greyed out. If you click on it again you get the same response. Close the dialogue and return later that day, Grey Fox will still be greyed out. The conversation runs only once.

However, there are occasions when you would like the NPCs to change their response according to set conditions. The clearest example of this is any conversation influenced by the disposition Mini-Game. In that case you might want conversations to be repeated. So why not always allow repeated conversations?

Mainly it prevents any script attached to a response being run more than once.

For the most part, this flag is left unchecked and repeated conversations are forbidden.

Allow Repeated Stages check box:

The vast majority of storyline quests work on the basic idea that each stage is a development that precludes the previous stages. Once you've done stage 10 then it is considered as executed. Any effects that the stage has, like adding a sword or displaying a message, are then ignored in future. Using script functions like **SetStage** 10 on an executed stage have no further effect. (Other than setting the stage counter to 10).

If this flag is checked, a journal entry will be displayed and the quest stage *result scripts* will be executed every time **SetStage** is called for a particular stage number.

Again, the vast majority of the time, we should leave this unchecked.

Add Icon Image:

This can be used to add an image file for the quest icon that is displayed in the player's quest log. You will need to have a custom made icon or an icon extracted from the BSA files.

Quest Conditions:

This creates a general set of conditions that influence all ***dialogue*** contained in this quest. A few quests have no limiting conditions. A few have multiple conditions.

But the vast majority use the condition :

GetIsPlayableRace == 1.

This is a script function. It checks that the Actor (NPC or Creature) involved in this scripts dialogue is from a valid playable race.

This means that invalid races like goblins don't greet you and have conversations. It's a kind of catch all condition that says; *ok let's keep conversations to Actors* that *can actually talk*.

I like Shadowmere, but it's going to freak me out if he greets me with "You're a sneaky looking sort' every time I go to ride him.

You can combine conditions to form **AND** conditions, that **all** must be met.
You can also use the **OR** flag to add **either/or** conditions.
This can also be combined with the faction setting to limit or remove general conversation topic from actors.

View Filtered Dialogue Window:

This brings up the filtered dialogue window. This is a **very useful** tool. It enables you to look at dialogue by Quest, by NPC and by Quest **and** NPC. In other words you can work out who says what, where, when and why?

Recompile All Results:

All results scripts (on ALL quests, not just the selected quest) are recompiled. Apparently this is a useful debugging tool. I avoid it like the plague myself. I've read too many horror stories about the effect of recompiling scripts.

Export Quest Dialogue:

This exports the entire quest's dialogue to a text file in the Oblivion folder. It exports it as a *tab delimited* text file, and can be hard to read. The same format can be used to import dialogue using the Data/Input menu. Tab delimited files can be imported into spreadsheets and databases, and be exported from them. Don't know any technical details and haven't tried to use this. It's just an observation.

Quest Stages Tab:

(I will use quest ID MS02 as an example here)

This is the nuts and bolts part of the quest. Firstly it is important to bear in mind that the *Quest Stage 10* shown here is not the beginning of the quest. Because this quest, *MS02*, was flagged as *StartGame Enabled***,** it has been running continuously from the first moment you began to create a character. Every five seconds the game has run the quest script and tried to follow its instructions.

Stage 10 is a discrete set of instructions. It has been quietly waiting for the right trigger to do its thing. So what triggers it?

Well before we see that, lets quickly tour the **Quest Stage Tab**.

Quest Stage Index:

- Each stage has an index number from 0 to 255.
- It is common practice to spread these in increments of **10**.
- In theory you could have a quest that has 255 stages but this is unlikely.

Why in 10's?

Because even the best planned quests, can have unexpected results which may require tweaking. Using 10, 20, 30 ... allows you to add an extra stage 15 or 25 in between. If you'd used the numbers 1, 2, 3..., you would have had to rewrite and reassign stages before you could fit in any new stage.

In this quest I'd put a small bet on extra stages being added in development. (Look at the index numbers.)

Quest Stage Items:

If you wish, you can add a **stage item** to your stage. This is a set of instructions to the game. It says when I reach stage *XX* , I want the program to do all this. If it suits your quest you can leave this blank. This might be the case if all you want to do at this stage in a quest is to enable some new dialogue. Each stage *can* have *more* than one *"stage item"*.

Each Stage Item can, but does not need to include:

- A Journal Entry,
- A Result Script
- A Set of Conditions

When the stage is activated, usually by a **SetStage** function, the game carries out each stage item in order and then internally marks it as done. (See Repeating Stages.) It updates the Journal if appropriate, and runs any result scripts, provided the conditions are met.

Multiple stage items are often conditionalised so that only one is actually run. This means only one update is sent to the player's log when a stage is set.

Log Entry:

This shows the text that is displayed in the player's quest log.

Result Script:

These script commands are run when the stage item is applied.

Conditions

The conditions must be true for the quest stage item to be applied.

Complete Quest check box:

If this box is checked, setting this stage will move the quest from the active to completed portions of the player's journal.

[*Note that a completed quest can still be running. You have to use a script command **StopQuest** to disable it completely*]

Export Quest Stages:

All the text and conditions of quest stages can be exported to a text file. This can then be spell checked and proof read. There is again an import facility on the File menu. (When I tried it - CTD! But that might just be me.)

Quest Targets Tab:

The Target Tab is used to set a map marker which appears in the world and mini maps to show the location of quest targets like Buildings and NPCs. Depending on the conditions specified for the target, multiple quest targets can appear at the same time. All targets (for the player's active quests) which pass their conditions will be displayed on the player's compass and map.

Target Ref:

A target must be a persistent reference.

Conditions:

The developers usually limit the targets selected by quest stages, but any valid conditions can be used.

Select Reference in Render Window:

The reference can be selected from the render window. All persistent references show a white cursor. Non-persistent references show a red cursor.

Cell & Ref:

Alternately, the reference can be selected by using the cell list and the list of references in that cell. Only persistent references are shown in the list.

Compass Markers Ignore Lock check box:

If checked, the path to the quest target will ignore locked doors (when showing the player the closest load door to reach the target). Otherwise, the path will attempt to avoid locked doors if possible.

So how did this quest start?

The answer is in the **dialogue**.

All the other tabs deal with *talking* in one form or another. The

Combat
Persuasion
Detection
Service and
Misc

tabs are all special dialogue topic tabs.

Topics Tab

When you click on an NPC and activate them, the program cross references the NPC's dialogue to decide what the NPC will say to you.

Every dialogue begins with a:

GREETING:

This topic is used when the player enters dialogue with an NPC. The nature of the greeting is contextualised by a series of conditions. (What progress you have made in quests, your faction, your race, your class etc.)

This is spoken as you enter the dialogue menu by activating the NPC. The Character animates, to turn and face you, and delivers his GREETING.

What appears in the left hand side in the menu is determined by context based statements attached to topics.

Some, like Rumours, are pretty much universal.
Some are limited to a few NPCs in special circumstances.

You can create new dialogue topics as you need them by editing and adding topics. (A word of caution here: Take great care to set conditions on dialogue to preventing it being said by every character.)

We can link topics to form a structured dialogue by using the *Link from*, *Topics* and *Choices* boxes on the right hand side to extend the range of Topics that appear for a given NPC.

Any dialogue spoken outside the topic menu in the game is covered by the other tabs - in particular, the CONVERSATION tab.

Conversation Tab

A conversation consist of a series of snippets of speech or, as the designers refer to them, **infos** that are played between two or more **NPC's**. All random conversations use infos stored in topics listed under this tab.

For example:

GOODBYE:

This topic is stored on the Conversation Tab, NOT the Topics Tab, and is played when the TOPIC menu is closed in the game.

The structure of the random NPC conversations is:

HELLO:

All random conversations start with a HELLO. The starting HELLO must have "Link From" left blank to indicate that it can start a conversation.

INFOGENERAL:

This topic is unusual because it appears both in random conversation and as the "Rumours" topic in dialogue with NPCs. When you ask an NPC about a Rumour, he will pick an info from the INFOGENERAL topic and then "keep" that rumour for 24 hours (so if you ask him again during the same day, he will continue to tell you the same rumour).

GOODBYE:

By default, all random conversations will end with a GOODBYE unless they are ended by another info with the "Goodbye" flag checked.

The Scripting Window:

Scripting is a whole new world of stress for the uninitiated newbie. It seems daunting at first trying to figure out what all the functions are for, unscrambling and understanding commands and how they go together to make something happen, and the joy of watching your first script produce unexpected, unintentional, and indecipherable results (if any). I will not cover all the console functions. I will cover scripting in three parts; first the ‘commands’ used in a line to produce a result, second the syntax, or how to put them together with a function to produce a result, and third, functions and what they do (and don’t do!). I will commonly refer to commands and functions within a script as ‘code’ in this section. There are quite a few conventions to keep in mind when scripting:

Keep in mind:

While I have tried to include as much information as possible this section is intended to give even a newbie a good base to learn scripting, and as a reference for anyone else. But this does not cover everything. To do that would take much longer, add several hundred pages to this work.

Making Scripts:

- I refer to any script that when written up in the CS as ‘big’ if you can’t see all the code at one time. If you have to scroll up or down to see any part of the code, I consider it big. Anything else is ‘small’ and easier to work with.

- The Scripting language is not case sensitive (Capital and lower case). However, if you choose to follow the convention used by Bethseda (which is to capitalize each word within a function. Ex: GetGameHour), or choose to make everything lowercase, stick to it. I suggest using the convention Bethesda uses, it makes reading a script easier, and picking out the functions and commands you are looking for. **BUT** whatever you choose to do, **stick to it**. It makes for cleaner scripts that are easier to read and understand. This is admittedly minor, but will save you some grief.

- Spaces are usually a good thing. When you compile a script a space is ignored; so often you can leave a space out with certain commands BUT (and this is a big but, kinda like mine); you may experience problems with the script if you do so. Thusly, if I mention it is a good idea to use a space somewhere, DO IT. In general it is a good idea to use spaces between all commands and functions. This is a big grief saver.

- Use a comma after a function and between parameters (numerical values). Scripts will usually compile and work if you forget to use a comma, but you might have errors if you don't. Rare, but better safe than sorry. With some functions you must use a comma afterward, usually those that involve filename use.

- Set and use 'do once' conditions. Because a script processes every frame, anything you have a script do, it will do 20 or more times each second. You probably don't want a script that acts as a trap on a chest to zap a player 20 times in 1 second with damage, every second, to infinity. Declare a variable called 'doonce', check if the variable is equal to 1 or not, then after a function processes you only want to happen once, add a line that sets the doonce variable to 1, next frame the script should check the doonce variable before the function, and since set to 1, skips the function. It should look like this:

```
If ( doonce == 1 )
   Return
Else
   [some function]
Endif
```

  Learn to use a do once statement in your scripts.

- Write out your script on paper first, at least a draft. I will put to paper a rough draft, maybe leave out a lot of the code even, just so I have a good working idea of how easy/hard the script will be, and identify any problems that may arise. Some do this, some don't. I usually don't write much code, just the concept of each section of a script. This can help you when working with big scripts.

- Divide your script into sections, and then fill it in with the necessary code. When I write out the concept of a new script I divide it up into sections. Each section of the script is accomplishing something for the script as a whole. One section may only be one or two lines, maybe compute some numbers for a variable, or maybe nothing more than checks for data. Whatever. This will help you make sure all parts of a script are doing what they are supposed to do when writing it out, and when debugging it. Trust me, for a big script; it's a grief saver.

- Use the semicolon (;) to comment on what you are doing in each section of a script. Especially if you are just learning how to script. This will keep you focused on what you are trying to do with the code of that section. In a big script it can be easy to loose track or forget what you were trying to do exactly with a given section.

- To save a script you must click the 'save' button. If there are no errors in your code it will save your script. If there are errors, it will not save and will alert you to the fact something is wrong (though not necessarily what). You will get an error statement when you try to compile for the first error found. So you may have to click save, fix an error, hit save again, fix another error, hit save again, etc. Just because you don't have an error when you compile a script doesn't mean it will work, the engine just checks to make sure there are no syntax errors. Your script might still be crud (and this is why we test our plug-ins people).

How Scripts Work:

- There is a small dropdown box located at the top of the Script Window saying **Script Type**. If your script is for a Quest make sure it is set to Quest. If it's for an Object, set it to Object. And the same for a Magic Effect.

- **Scripts run once every single frame** (you remember what frames are right?). This means a script may execute or process 10, 20, 40, or more times every second. Also, frame rates vary from computer to computer, and second by second on any single computer. Keep this in mind.

- There are two types of scripts: Local and Global. Local scripts are any script that's attached to an object. A local script will only affect the cell it is in, but will and/or can affect anything in that cell (you allow it to). Local scripts will automatically start the moment the PC enters the cell (in an exterior cell, the moment that cell is loaded, even if the PC isn't 'in' it yet). Global scripts must somehow be started, are not attached to anything, but process regardless of what cell they are in (according to their code). You must use another script to start a global script or to stop it (whether that other script is a local or a global is up to you). Some global scripts start at the beginning of the game to set global variables and set how the game is to run (called startup scripts). Don't mess with these, trust me, just don't.

- Local scripts, since they are attached to an object, usually affect only that object with their code unless you specify differently (within that script).

- Global scripts, since they are not attached to any object, must have specified within their code, what they are affecting with any function stated.

- Scripts use variables to *hold* or *remember* data (and you thought algebra would never come in handy back in school). The variable is a single word (or several together without spaces OR connected with an underscore). They *hold* data, or numbers. You must list (or *state*) variables you will use in a script at the start of a script before any code (but after the `Begin` command). This is referred to as '*declaring*' a variable in a script. Most of the time, a variable will have a value of -1 (negative one) until you assign data in the script.

- Variables come in three flavors; Long, Short, and Float. You must state which type a variable is supposed to be when you declare it. I have yet to see a Long variable declared in a script, but you can.
- **Long variables** can be assigned a value anywhere from -2,147,483,648 to 2,147,483,647.

- **Short variables** can be assigned a value anywhere from -32,768 to 32,767.

- **Float variables** can be assigned a value anywhere from 3.4E +/- 38. What does this gobbledygook mean? It means the variable can accept a numerical value that has a decimal in it, with up to seven digits after the decimal.

- Once a value is assigned to a variable it will always have that value until changed, no matter if a local or global variable, what cell you are in, exiting the game, or turning your computer off.

- In a local script, you can pretty much make up any name for a variable you want, the variable will only work with that script. Try to make it memorable, and easy to identify what data the variable holds (ex: if you use a variable to hold the current Strength of the PC, name the variable 'currentstrength' or 'currstr'). Try not to accidentally use function names, and don't use punctuation. Both will really screw things up.

- The game already has some global variables declared. They are hard coded into the game; you cannot change them (without serious game breaking repercussions). You can use them in a script though. Some you don't have to declare, and some you do.

The Commands:

These are the words and symbols used in a line of the script to state how the script processes and what is affected by it.

ScriptName:

This is the command that states the name of the script. You put the name of your script on the same line after this command. The script name cannot have any spaces. Thus either of the following is a valid way to write the first line in a script:

`ScriptName CatNightEye` OR `ScriptName Cat_Night_Eye`

This is a no-no:

`ScriptName Cat Night Eye` OR `ScriptName CatNightEye`

Begin:

The begin command has to be before any function. Example:

```
ScriptName AexampleScript

Begin GameMode

        [Some Function]
```

After begin you must always state when you want it to begin.

End:

This goes on its own line at the very end of a script. This tells the game no more code for this script; it's done for this frame. It does not mean the script stops getting processed (check above bullets).

Semicolon:

Use a semicolon (;) to add text. Any text you place after the semicolon on the same line will be ignored when the script is compiled and processes in game. This doesn't mean it's deleted; it stays where you put it, and it's just ignored. You cannot use the 'enter' key for a carriage return! Only the text placed after the semicolon and on the same line is ignored. If the text wraps around to the next line, that's ok. Thus, this is ok:

```
Set GameHour to 10   ; set the current game hour to 10
```

But not this:

```
Set GameHour to 10     ; set the current [carriage return]
        [tab][tab][tab]; strength for testing against [carriage return]
        [tab][tab][tab]; item weight
```

Note: *You cannot put anything on a line above this command, except text after a semicolon. Don't do this though, just put any descriptive text below this command.*

If:

You use this command to check if some aspect of the game meets a condition you have stated. Thus:

```
If ( GameHour == 10 )  ;checks for the Game Hour exactly equal to 10
```

The `if` command checks to see if the function statement following it is true or false. If true, it will process any functions listed below it until another command statement is reached. If false, if will skip any functions until another command statement is found below it.

ElseIf:

Use this below an `if` statement to check for more than one condition. Thus:

```
If  ( GameHour ==  10 )       ;checks for Game Hour equal to 10
       [some function]
ElseIf ( GameHour == 5 )      ;checks for Game Hour equal to 5
       [A different function]
```

Think of the `ElseIf` command as similar to a Boolean 'and' or 'or'. It otherwise works just as does the `if` command.

Else:

Use this after any `if` or `ElseIf` commands to check a final or 'catch all' threshold or condition. Thus:

```
If ( GameHour == 10 )         ;checks for Game Hour equal to 5
       [Some function]
ElseIf ( GameHour == 5 )      ;checks for Game Hour equal to 5
       [A different function]
Else                                 ;if Strength is not equal to 30 nor 40
       [Entirely different function]
```

Think of the `else` command as 'if none of the above'. If no `if` or `ElseIf` conditions above it return true, this command will process any functions listed below it until the another command is reached.

Set/To:

Use this pair to assign a value to a variable:

```
Set GameHour To ( GetStrength.player )
```

Endif:

You must use this command to finish any (set of) `if` statement(s). The script will not compile without all If/ElseIf statements ending with an `Endif` command. Thus:

```
If (GameHour == 10 )          ;checks for Game Hour equal to 10
       [Some function]
ElseIf (GameHour == 5 )       ;checks for Game Hour equal to 5
       [A different function]
```

-

```
Else                                  ;if GameHour is not equal to 10 nor 5
        [Entirely different function]
Endif
```

You can also embed `if` statements within other `if` statements. The indented `if` statement is said to be embedded or nested within the first `if` statement (There is reported to be a maximum of ten embedded if statements within a singular if statement. I do not know, I've never tried to see if this is so). Thus:

```
If ( GameHour == 10 )         ; checks for Game Hour equal to 10
        If ( GameDay == 20 )  ; checks for Game Day and Game Hour both
                                      ; Equal to 20
                [Function]
        Endif
        [Some function]               ;if Game Hour is 20, but Game Day is not 20
ElseIf ( GameHour == 24 )     ;checks for Strength equal to 24
        [A different function]
Else                                  ;if Game Hour is not equal to 10 nor 24
        [Entirely different function]
Endif
```

This is a simple `if` set of statements. Note the structure. The `if` command comes first, then any `ElseIf` commands (optional, you may not need to use any. You can use more than one if you need to, quite a few in fact), then the `else` command (optional, you don't have to use this command, but useful when you only need to check for a few conditions out of many). Then the `endif` command to end the set. Notice how I indented the [function statement]. The function is embedded in the `if` statement and only gets processed when the `if` statement returns (or is) true. It is not necessary to indent, but is oh so useful when scanning through lines of code trying to understand what you have done (and/or done wrong).

Syntax:

A full stop (.) is used to designate the calling object, and the condition or function to apply to it. The calling object is usually the object_ID on the left side of the 'full stop'. The condition to check or function to apply is usually stated to the right of the 'arrow'. Thus:

```
object_ID.GetStrength   OR   Player.SetStrength 50
```

Note: *The object 'ID' of the PC is always* '`player`'.

The function can also be placed on the left and the object's ID on the right. Thus:

```
SetStrength.player 50
```

But not:

```
GetStrength.NPC_ID
```

-

Why? In the first you are applying the function to the PC, the numerical value can be placed after the object as so. But in the second you are calling for what that numerical value is, but cannot assign it to a variable with the given statement syntax. In the above statement you're telling the script to 'get' the Strength of the NPC but cannot tell the script what to do with it.

When you state an object's ID in a script, the game will look for the first such object_ID listed in it's database. Which means you might not get the object's reference you want the script to work with. This is why it's a good idea to give an object a unique name that you want to attach a script to. If an object has it's *References Persist* box checked, the object can be accessed by a script regardless of what cell either is in.

Condition Check signs:

== This is used to set a condition of 'is equal to'.
!= This is used to set a condition of 'is not equal to'.
<= This is used to set a condition of 'is less than or equal to'.
< This is used to set a condition of 'less than'.
>= This is used to set a condition of 'is greater than or equal to'.
> This is used to set a condition of 'is greater than'.

These are used as thus:

```
If ( player.GetStrength >= 30 )       ;if the PC's Strength is equal to or greater
than 30
       [function]                     ;do this
Endif
```

Note: *Almost always when you make an `if` statement you will need to use parentheses to encapsulate the condition check part of the. Whenever you use parentheses you should make sure to put a space on both sides of each parentheses. Don't argue, DO IT! Also make sure to put a space before and after the condition setting signs (i.e. `==`, `!=`, `<=`, `<`, `>=`, `>`). If you have to ask why, see the above bullet about using spaces in code.*

Mathematics:

`+` This is used for addition.
`-` This is used for subtraction.
`*` This is used for multiplication.
`/` This is used for division.

Math is done just like you learned in school. Thus:

```
Set variable_x to ( variable_y + variable_z )  or
Set currentstrength to (( variable_a + variable_b ) * (( variable_c - 10 ) / 5 ))
```

-

Note: *The rule about spaces applies to math symbols too (a space on each side).*

Note: *I have seen several statements saying that there is a maximum number of variables in math calculations that can be done on one line of code. Supposedly 20, I have not tried to exceed this to see if true, but recommend you split big calculations between several lines to avoid the 'rule of 20'.*

Let's review Commands. Each script must have a `Begin`, then code, and then `End`. You use `if` statements to test conditions. You can use `set/to` to assign numerical values to variables, or to assign one variables value to another variable. You use variables to hold these values within a script and/or to manipulate them. You must declare variables. Most variables start at a value of -1 before assigned a value. Use spaces. Here is a sample script (below) so you can see a complete format. The script begins, declares a variable, checks if this script has ever processed before, tests if Strength is under 40, then tests if Agility is under 40, and if so sets Strength to 40. If either Strength or Agility is over 40 then no changes are made. After the `if` statements, the variable is changed to a 1. The script then ends and starts processing again. Now, on the second run through the script the game gets to line 5, checks the variable's value which was set to 1 last frame. Since it is 1, greater than 0, the game skips the entire set of if statements, ends, and starts over again the next frame, ad infinitum.

```
ScriptName ASampleScript

short variable_one                          ;declared a variable

Begin OnActivate                            ;script starts processing here

if ( variable_one < 0 )                     ;has this script processed before?
   if ( player->GetStrength < 40 )          ;is the PC's Strength less than 40
       if ( player.GetAgility < 40 ) ;is the PC's Agility less than 40
          Player.SetStrength, 40            ;if not, change Strength to 40
       Endif
   Endif
Endif

Set variable_one to 1                       ;change variable_one to 1

End                                         ;script ends here
```

Note: *this script is just to show format. While it would work, it's not very efficient.*

The Functions:

You're still not ready to start scripting yet; you need to know about functions and what they do. There are many, many functions. Most are useful, a few have limited use, and several just plain don't work. I will list each function, but I will not be giving descriptions of what they do, maybe in a later version. (If you know a function that is not listed here please email me and tell me what it does) Let's get to the functions.

-

A

Activate

AddAchievement

AddFlames

AddItem

AddScriptPackage

AddSpell

AdvancePCLevel

AdvancePCSkill

Autosave

C

CanHaveFlames

CanPayCrimeGold

Cast

CloseCurrentOblivionGate

CloseOblivionGate

CompleteQuest

CreateFullActorCopy

D

DeleteFullActorCopy

Disable

DisableLinkedPathPoints

DisablePlayerControls

Dispel

DispelAllSpells

Drop

DropMe

DuplicateAllItems

DuplicateNPCStats

E

Enable

EnableFastTravel

EnableLinkedPathPoints

EquipItem

EssentialDeathReload

EvaluatePackage

F

ForceActorValue

ForceCloseOblivionGate

ForceFlee

ForceTakeCover

ForceWeather

G

GetActionRef

GetActorValue

GetAlarmed

GetAmountSoldStolen

GetAngle

GetArmorRating

GetArmorRatingUpperBody

GetAttacked

GetBarterGold

GetBaseActorValue

GetButtonPressed

GetClassDefaultMatch

GetClothingValue

GetCombatTarget

GetContainer

GetCrime

GetCrimeGold

GetCrimeKnown

GetCurrentAIPackage

GetCurrentAIProcedure

GetCurrentTime

GetCurrentWeatherPercent

GetDayOfWeek

GetDead

GetDeadCount

GetDestroyed

GetDetected

GetDetectionLevel

GetDisabled

GetDisposition

GetDistance

GetDoorDefaultOpen

GetEquipped

GetFactionRank

GetFactionRankDifference

GetFactionReaction

GetForceRun

GetForceSneak

GetFriendHit

GetFurnitureMarkerID

GetGameSetting

GetGlobalValue

GetGold

GetHeadingAngle

GetIdleDoneOnce

GetIgnoreFriendlyHits

GetInCell

GetInCellParam

GetInFaction

GetInSameCell

GetInWorldspace

GetInvestmentGold

GetIsAlerted

GetIsClass

GetIsClassDefault

GetIsCreature

GetIsCurrentPackage

GetIsCurrentWeather

GetIsGhost

GetIsID

GetIsPlayableRace

GetIsPlayerBirthsign

GetIsRace

GetIsReference

GetIsSex

GetIsUsedItem

GetIsUsedItemType

GetItemCount

GetKnockedState

GetLOS

GetLevel

GetLockLevel

GetLocked

GetNoRumors

GetOffersServicesNow

GetOpenState

GetPCExpelled

GetPCFactionAttack

GetPCFactionMurder

GetPCFactionSteal

GetPCFactionSubmitAuthority

GetPCFame

GetPCInFaction

GetPCInfamy

GetPCIsClass

GetPCIsRace

GetPCIsSex

GetPCMiscStat

GetPCSleepHours

GetPackageTarget

GetParentRef

GetPersuasionNumber

GetPlayerControlsDisabled

GetPlayerHasLastRiddenHorse

GetPos

GetQuestRunning

GetQuestVariable

GetRandomPercent

GetRestrained

GetScale

GetScriptVariable

GetSecondsPassed

GetSelf

GetShouldAttack

GetSitting

GetSleeping

GetStage

GetStageDone

GetStartingAngle

GetStartingPos

GetTalkedToPC

GetTalkedToPCParam

GetTimeDead

GetTotalPersuasionNumber

GetTrespassWarningLevel

GetUnconscious

GetUsedItemActivate

GetUsedItemLevel

GetVampire

GetWalkSpeed

GetWeaponAnimType

GetWeaponSkillType

GetWindSpeed

GoToJail

H

HasFlames

HasMagicEffect

HasVampireFed

I

IsActionRef

IsActor

IsActorAVictim

IsActorDetected

IsActorEvil

IsActorUsingATorch

IsActorsAIOff

IsAnimPlaying

IsCellOwner

IsCloudy

IsContinuingPackagePCNear

IsCurrentFurnitureObj

IsCurrentFurnitureRef

IsEssential

IsFacingUp

IsGuard

IsHorseStolen

IsIdlePlaying

IsInCombat

IsInDangerousWater

IsInInterior

IsInMyOwnedCell

IsLeftUp

IsOwner

IsPCAMurderer

IsPCSleeping

IsPlayerInJail

IsPlayerMovingIntoNewSpace

IsPlayersLastRiddenHorse

IsPleasant

IsRaining

IsRidingHorse

IsRunning

IsSneaking

IsSnowing

IsSpellTarget

IsSwimming

IsTalking

IsTimePassing

IsTorchOut

IsTrespassing

IsTurnArrest

IsWaiting

IsWeaponOut

IsXBox

IsYielding

K

Kill

KillAllActors

L

Lock

Look

LoopGroup

M

MenuMode (Function)

Message

MessageBox

ModActorValue

- ModAmountSoldStolen
- ModBarterGold
- ModCrimeGold
- ModDisposition
- ModFactionRank
- ModFactionReaction
- ModPCAttribute
- ModPCFame
- ModPCInfamy
- ModPCMiscStat
- ModPCSkill
- ModScale
- MoveTo
- MoveToMarker

P

- PayFine
- PayFineThief
- PickIdle
- PlaceAtMe
- PlayBink
- PlayGroup
- PlayMagicEffectVisuals
- PlayMagicShaderVisuals

PlaySound

PlaySound3D

PositionCell

PositionWorld

PreloadMagicEffect

R

RefreshTopicList

ReleaseWeatherOverride

RemoveAllItems

RemoveFlames

RemoveItem

RemoveMe

RemoveScriptPackage

RemoveSpell

Reset3DState

ResetFallDamageTimer

ResetHealth

ResetInterior

Resurrect

Rotate

S

SameFaction

SameFactionAsPC

SameRace

SameRaceAsPC

SameSex

SameSexAsPC

Say

SayTo

ScriptEffectElapsedSeconds

SelectPlayerSpell

SendTrespassAlarm

SetActorAlpha

SetActorFullName

SetActorRefraction

SetActorValue

SetAlert

SetAllReachable

SetAllVisible

SetAngle

SetAtStart

SetBarterGold

SetCellFullName

SetCellOwnership

SetCellPublicFlag

SetClass

SetCombatStyle

SetCrimeGold

SetDestroyed

SetDoorDefaultOpen

SetEssential

SetFactionRank

SetFactionReaction

SetForceRun

SetForceSneak

SetGhost

SetIgnoreFriendlyHits

SetInCharGen

SetInvestmentGold

SetItemValue

SetLevel

SetNoAvoidance

SetNoRumors

SetOpenState

SetOwnership

SetPCExpelled

SetPCFactionAttack

SetPCFactionMurder

SetPCFactionSteal

SetPCFactionSubmitAuthority

SetPCFame

SetPCInfamy

SetPCSleepHours

SetPackDuration

SetQuestObject

SetRestrained

SetRigidBodyMass

SetScale

SetSceneIsComplex

SetShowQuestItems

SetSize

SetStage

SetUnconscious

SetWeather

ShowBirthsignMenu

ShowClassMenu

ShowDialogSubtitles

ShowEnchantment

ShowMap

ShowRaceMenu

ShowSpellMaking

SkipAnim

StartCombat

StartConversation

StartQuest

StopCombat

StopCombatAlarmOnActor

StopLook

StopMagicEffectVisuals

StopMagicShaderVisuals

StopQuest

StopWaiting

StreamMusic

T

ToggleActorsAI

TrapUpdate

TriggerHitShader

U

UnequipItem

Unlock

V

VampireFeed

W

Wait

WakeUpPC

WhichServiceMenu

Y

Yield

Oblivion Script Extender:

The Oblivion Script Extender, or OBSE for short, is a modder's resource that expands the scripting capabilities of Oblivion. A lot of people have trouble loading up OBSE for the first time, you have to open a command prompt window, navigate to your oblivion install folder, and type "obse_loader -editor". This is quite confusing; personally, I have no idea how to do this at all. So instead, I will tell you a very simple way; but requires you to download a program called Fast Command Prompt.

Running the CS with OBSE:

Well first I had better begin by telling you what Fast Command Prompt (FCP) is. FCP is a very small (Below 200kb) and easy to use program (If you wish to download go to a search engine website and type: Fast Command Prompt Download.). It allows you to R-Click on any folder on your HD and click Launch Command Prompt. This will instantly bring up a command prompt window that has automatically navigated to this folder, very helpful. Once you R-Click on your Oblivion folder (Default Directory: C:\Program Files\Bethesda Softworks\Oblivion) click on Launch Command Prompt, then all you need to type is obse_loader –editor. This will load up the editor with the OBSE functions enabled.

Writing your first OBSE script:

This will be included in a future update.

The Magic Effects Window

This window allows you to modify the visual effects seen when spells are cast. Highlight an effect and the appropriate visual effects are listed in boxes and drop down menus to the right. These are explained below. Take care and give consideration to any decision to modify anything in this window as it could easily unbalance gameplay far more than intended.

Name:

This is the in game name of the spell.

School:

This drop down menu lists each spell effect school. Use this to change what school a spell effect is associated with (and what skill it is based on).

Base Cost:

This is the cost in Magicka points as a base that the game uses to calculate the cost of casting a spell. Remember this is just a base; other factors modify the cost.

Resist Value:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Assoc. Item:

This is used for summoning creatures or bound weapons; simply click the dropdown box and choose what you wish to be able to summon.

Effect Icon:

This is the icon shown at the bottom of the screen in game when a spell is cast using this spell effect.

Flags:

Spellmaking check box:

If this box is checked this spell effect will be included on the list of effects the PC can use to create a spell with.

Enchanting check box:

If this box is checked this effect will be included on the list of effects the PC can use to enchant items with.

Hostile check box:

If this box is checked this effect is classed as hostile; if you cast it on somebody you will get in trouble with the guards.

Detrimental check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Recover check box:

If this box is checked then the effect heals or fortifies the PC or NPC.

Magnitude % check box:

If this box is checked then the effect is a weakness or resist effect.

FX Persist check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Self check box:

If this box is checked then the effect is cast upon the PC or NPC; depends who casts it.

Touch check box:

If this box is checked then the effect is cast upon a touch of another creature or NPC.

Target check box:

If this box is checked then the effect is cast as a target spell; this means the spell acts like you are throwing a rock at somebody, but in a straight line.

No Duration check box:

If this box is checked then the effect is not a Duration effect.

No Magnitude check box:

If this box is checked then the effect is not a Magnitude effect.

No Area check box:

If this box is checked then the effect is not an Area effect.

No Ingredient check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Use weapon check box:

If this box is checked then the effect is a bound weapon effect.

Use Armor check box:

If this box is checked then the effect is a bound armor effect.

Use Creature check box:

If this box is checked then the effect is a summon creature effect.

Use Skill check box:

If this box is checked then the effect is used for a skill; whether It be a fortify skill or absorb skill, if its something to do with a skill; its checked.

Use Attribute check box:

If this box is checked then the effect is used for a attribute; whether It be a fortify attribute or absorb attribute, if its something to do with a attribute; its checked.

No Recast check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

No Hit Effect check box:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Counter Effects:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Sounds:

Casting Sound:

This is the sound played when either PC or NPC casts the spell.

Bolt Sound:

This is the sound played when the spell is moving or doing what it is supposed to do.

Hit Sound:

This is the sound played when the spell hits the target, or cast upon yourself or an NPC.

Area Sound:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Constant Effect Enchantment Factor:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Constant Effect Barter Factor:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Visual Effects:

Effects NIF:

The NetImmerse effect file; this appears when the spell is cast; it could be a fireball etc.

Effect Shader:

The shader of the casted spell effect, you shouldn't modify this unless you know what you are doing.

Enchant Effect:

At the moment I do not know what this does. If you do, please email me so I can place it into this manual.

Projectile Type:

This can be set to Ball, Bolt, Fog or Spray; this control what kind of projectile is cast when the spell gets cast. If it were a fire*ball* you would set it to ball.

Projectile Speed:

This toggles how fast the chosen projectile travels through the air.

Light:

This controls what light the effect produces when cast.

Description:

This box shows the description for this magic effect.

Chapter 5 - Saving and Testing Your Mod

You finished you mod. Your plug-in is ready to be uploaded for others to enjoy. Did you test it? Did you clean it? What do I mean??… You got a little more work to do before you upload your mod. This chapter I will help you figure out if your mod is dirty and hopefully get as much testing done as possible, without causing problems for your mod.

Creating a Mod:

It's so simple. The CS is so easy to use, right? Well, it actually is, despite how it may seem at first (ever worked with a professional auto-cad program?). When the CS first starts you have a blank world, almost no objects, no races, factions, etc. The pull to create a new world is tempting for some (a Total Conversion). For others it is the chance to tweak the game to make it easier, or more difficult. To others it is the opportunity to expand an area of the game, basically, to make it more fun. The best thing you can do when starting is to take your idea, or vision, and write it down. Give it some focus. If your vision is just to create a new race, or make a house, that's pretty simple and you probably don't need to write out a mission statement. But if you're going create a mod with new areas, NPCs, quests, items, etc. Write down the storyline, this will help you stay on track as to what you're doing, and why. If the idea changes, fine, go with it. But this will also keep your idea pure. If while working on one mod you get the idea for another, write it out and work on it separately. Unless you've got some experience creating mods, try not to combine ideas. Keep them separate, and if one would work well in the other you can always combine one with another. Keeping them separate can also help to keep the debugging process simpler. Working on just one project at a time also keeps you on track for completing it. If you work on three different mods at once, none will be done till you've completed all three. These are just suggestions, everyone has their own style of doing things.

What is your base .esm(s)?

Will your mod be dependent on only Oblivion? Maybe Oblivion and KOTN? SI? What patches do you have installed? Make sure you decide and note this in the readme.txt. Save some grief for those who want your mod, download it, and discover they can't play it because you didn't mention that you used Oblivion.esm and KOTN.esm as the dependent files for your plug-in, but they only have SI so far. If you want to make a name for yourself don't make it a bad one.

Take notes, lots of notes:

Use a notebook to keep track. Write down any objects you create by ID, the changes you made (if any), what cell you put it in, or even the grid. Created some interiors? Keep a running tab of all objects used for that interior, scripts you've attached to objects (and what objects), reason for that interior (it's purpose). Making a quest? Write out a timeline of events and things that have to happen (and those that could happen), especially if you have more than one event taking place during the quest.

Test you plug-in:

Most people do this; mostly because they want to play the great mod they created. But you have to do more than that. You have to explore every part of the mod you create. If you place an NPC in your mod with dialog, click on every response to verify you get every response you gave them. If you write a script, get it to activate and test every possible condition you wrote into it. If you create objects, use them and test them under every circumstance you can think of. This is how you find bugs, and how others will find them if you don't when they play your mod. This takes time too, which is why we hail beta testers.

User created objects:

Plan on using objects created by someone else you found in a mod you liked? Did you get permission to do so from the plug-in's author? The information on how to contact an author should be included with any plug-in readme.txt file. If one isn't included, where ever you downloaded the plug-in from should have had some kind of statement or at least the name of who submitted the plug-in. You should always make every attempt to get permission to use anything someone else created. If you don't and that author takes exception you could be sued. Plus, that's just showing respect for the work someone else did. Most are willing to let you 'borrow' their work if you just ask. Are you including new objects personally created by someone? Do you know how to find and include the correct `.nif`, `.kif`, `.tga`, `.dds`, or `.bmp` files in the package to upload? This could be a

---

problem. If you forget to include the files that didn't come with the game, your mod won't work right.

Compress the plug-in:

Always compress the plug-in and any associated files needed by it. This makes it smaller so it is easier to upload, and also download. What utility you wish to use is up to you. The help feature of the utility should tell you how to create a zipped file and how to add files to it.

Beta Test it:

Send it to some friends or find someone to do a test of your plug-in at a web site's forum. There are always people willing to beta test a plug-in. They can let you know what bugs and problems they find that you missed. If you just create a simple plug-in with a few changes or things added you probably don't need to go through all this to verify that it works correctly and did everything right. But it won't hurt if you do. These are things you can do to keep your plug-in clean and bug free, and the bigger it is, the more things you do in it, the greater the chance of problems. Why go through all the work you did only to have people upset because it's buggy, or unplayable. Avoid the bad karma, do the little extra time and effort to make sure your mod is good to go.

What's with the little * thingy?

It's called an asterisk. What is does though is let you (or anyone) know what you have changed in your plug-in. If you see any object's ID name with an asterisk next to it, it means at some point you modified or at least did something to that object. If you didn't make a change, but the object has an asterisk, congratulations, you have a dirty plug-in. How to fix? Make a note of any such objects you know you don't want any changes to, and clean it out by using the **Details** button on the Data window, or use a utility if you wish. Read on below under the *Cleaning the Unclean* section.

Note: *Only the active plug-in will display an asterisk next to objects that have been altered, but all objects will be listed in the Objects Window for all plug-ins loaded, active or not.*

Cleaning the Unclean:

Clean your mod. Sounds like something your mother might say. But it needs to be done. A dirty plug-in is usually larger than necessary (as in file size). This means it takes longer to download, and a big file may be a factor for some choosing what to download. Someone who isn't diligent enough to clean a plug-in most likely isn't diligent enough to debug either (that's my opinion though), and we all love buggy games, don't we

---

(**cough, cough*... daggerfall... *cough**). There are several utilities that can be used to help clean a plug-in (see Chapter 6), or you can use the CS itself.

To Clean:

Start the CS, *do not load your plug-in*. Open the Data window and find your plug-in, highlight it. Do not click on any other files. Click on the 'Details' button at the bottom of the window. This will bring up a new window. This window allows you to see all the changes you've made to your mod. On the far left is a column that states what the changed object is. The fourth column shows the object ID. The fifth column is the offset, if you don't like to hex edit files, don't worry about it. If you see an object you do not want in your mod, or an object you didn't mean to make a change too (a spec of dirt I guess you could say), highlight the object you wish to remove from your mod. Now tap your 'delete' key. The CS will prompt you, stating that this object's *change* will be *ignored* and *not loaded* if you continue and load the plug-in, effectively deleting this change. For this to work you MUST load your plug-in after marking all objects you want purged, and then save it! This is important! This is how you *clean* your mod of mistakes you've made by making changes to things you had not meant to.

The first column is your clue as to what an object is. Some of these are obvious. Here is a list and what they refer to:

ACTI – Activator.
ALCH – Potion.
AMMO – Ammunition.
ANIO - Animated object.
APPA – Apparatus.
ARMO – Armor.
BOOK – Book.
BSGN – Birthsign.
CELL – Cell.
CLAS – Class.
CLOT – Clothing.
CLMT – Climate.
CONT – Container.
CREA – Creature.
CSTY - Combat style.
DIAL – Dialog.
DOOR – Door.
EFSH - Effect shaders.
ENCH – Enchantment.
EYES – Eyes.
FACT – Faction.
FLOR – Flora.
FURN – Furniture.

GRAS – Grass.
GLOB - Global Variable.
GMST - Game Setting.
HAIR – Hair.
IDLE - Idle animation.
INGR – Ingredient.
KEYM – Key.
LIGH – Light.
LVLC - Leveled Creature.
LVLI - Leveled Item .
LVSP - Leveled Spell.
LTEX - Land texture
MGEF - Magic effects
MISC - Miscellaneous item
NPC - Non-player character
PACK – Package
QUST – Quest
RACE – Race
REGN – Region
SBSP – Subspace
SCPT – Script
SGST - Sigil stone
SKIL – Skill
SLGM - Soul gem
SOUN – Sound
SPEL – Spell
STAT – Static
TREE – Tree
WATR – Water
WEAP – Weapon
WRLD - World space
WTHR - Weather

I believe I have listed almost all, if not all.

The second column shows objects that are part of an `.esm` file that has been deleted in your plug-in. You can't actually delete an object from an `.esm` file, so the object is still present in the `.esm`, but will always be shown that you deleted it in your plug-in (deleting *objects* from an `.esm` is not a good idea, deleting *references* is better. Try to avoid deleting objects that are part of an `.esm`.). When your plug-in is loaded by the game it will not load this object. Objects created within a plug-in, not part of an `.esm`, are truly deleted and do not show up in this column.

The third column shows those that you wish to 'ignore' (delete). These are the objects and changes you wish to remove from an `.esp`.

Note: *References cannot be deleted from here. You do that by loading an* `.esp`*, and then opening a cell where the reference is placed, select it in the Render Window, then click delete. Or you could just delete it from the object window (if user created, not part of an* `.esm`*).*

Respecting things other people make:

If you wish to use something done by someone else, it's probably ok. But show some respect. They worked hard to make it, and want some recognition for having done so. Don't just add something you found in another plug-in to yours off handedly. Know who made the plug-in, and make an effort to contact them and ask for permission to include their work in your own. Often you will find a readme.txt file that will state that the author of a plug-in will allow this without contacting them. Usually there are certain limitations, such as including giving them credit for what you are including in your plug-in. Some may want you to do more. This is important, for while many may be willing to overlook a mod that included an object they created, some may not. There are plug-ins floating around the net that contain objects created by professionals using expensive software. If you don't credit them for their work and you use it in your plug-in, they could take legal action against you. I admit, I have yet to hear of a single case of this. But it's also a matter of respect. You may wonder why I included this section. Because I've found plug-in readme.txt files stating they got an object from this plug-in called 'something' and another object from some website they can't remember the name of. Or they used some scripts they liked from another mod without specifically naming the plug-in or its author. Simply put, I think that's bull. Show some respect, and give credit where it's due.

Thinking of NPCs Names:

One of the worst things when making a mod, a total conversation, is thinking of names to give to your NPCs, my recommended way is using a name generator; preferably this which is a free and easy to use tool.

Making a ReadMe:

You know you have to include a readme in with your mod, but how do you make it? This small template should give you an idea, you are free to use this and make a readme for you mod; no credit needs to be given.

Elder Scrolls IV: Oblivion
TemplateMod

Author: TemplateAuthor

Version: V1.0

Requirements:

Requires Official Oblivion Patch 1.2.404

---------------
Description:
---------------

TemplateMod adds a small buyable house just outside of Chorrol near the stables. The house costs 3000 gold and can be purchased from the count of Chorrol.

--------------
Installation:
--------------

Extract all the files from the ZIP into your Oblivion\Data director.

Activate TemplateMod.esp file from Data Files in the Oblivion Launcher.

---------------
Uninstalling:
---------------

Delete the following file:

TemplateMod.esp

-------------------
Version History:
-------------------

1.0
- Initial Release:

----------
Credits:
----------

Thanks to TemplatePerson for providing help about scripting.

----------
Contact:
----------

Contact me at:

Template@TemplateMod.com

Chapter 6 - Utilities

In this chapter I will try to give some insight into some utilities I have found, what they can be used for, usefulness, and maybe a couple tips to using them. This will not be a tutorial on how to use any of these. This will be just some basic information. By including this chapter I am not in any way endorsing any of these utilities, merely providing some information about them. Make your own judgments about whether they would be of any use to you, and if you should download and use them.

TES4 Plugin Utility:

The TES4 Plugin Utility has many useful features, such as merging mods, creating silent MP3 voice files and more!

BookGen (Book Generator utility):

Write the text of a book in HTML or text format, and then use to create the book for in game use.

BSA Commander:

This can be used to view, unpack, or repack a .bsa file. Even pack a new .bsa file.

Markov Name Generator:

With this program, you can generate new character names quite easily and the new names tend to sound like the normal race names.

NifSkope:

NifSkope is a tool for analyzing and editing NetImmerse files, it has many useful features; defiantly worth a download.

Oblivion Mod Manager:

Oblivion Mod Manager, as its name suggests, is a utility for managing oblivion mods. It can just be used as an enhanced version of the data files selector on the oblivion launcher, with sorting and load-order reordering capabilities, but there are a lot more features which can only be used when mods are packed as omod files.

Chapter 7 - Tutorials

In this section I will be providing tutorials. I will be getting these of various websites and giving the original authors credit and I will occasionally write one. These tutorials will cover the more difficult sections of the CS.

Making a Quest Tutorial:

This is what we will do with this tutorial:

- Add dialogue to an NPC.
- Making the journal update by picking up an item, or talking to an NPC.
- Receiving an item (key) from an NPC, so you can find the item he lost.
- Receiving a reward from an NPC, for giving that item back.
- Add quest markers, so the player knows where he has to go.

Before we start, I want to make sure you know a few basic things, which I won't explain in this tutorial.

- Being able to make new NPC’s and items by changing their ID.
- Being able to place such NPC’s and items in the cell you want.
- Being able to create a new container (chest), and place an item in it.

And a tip before we start: *Save often*. The construction set often crashes when you do something wrong, or even when you do something right! So save often!

Making the quest-giving NPC:

All right, let's start with making the items we need. Add a new npc to a cell. Be sure to make a new Unique ID.

This is wrong:

NPC1
MyNPC
Quest

This is right:
PovtutAlex1 (Pov stands for Povuholo, that's me. Placing that in front of all your ID's won't just make the ID unique; it will also make it easier to find in the object list. Tut stands for Tutorial, which is the name of my mod.)

Of course this is just how I do it, you should make your own ID's. Just make sure they are unique, we don't want anyone else use the same ID for his/her quest, which will probably prevent both mods from completely functioning.

So add the new NPC (mine is called Alex) to a cell. Let's say the Imperial City Market District. D-Click on him, tick the box Persistent Reference and make a new reference ID located at the top of the window. Mine is PovTutAlexRef1.

Adding the quest related items:

Make a new key with a new ID… My key ID is PovTutAlexKey1. Now, choose what item Alex lost. His own iron armour, perhaps? Go to the armour tab and take the iron armour, change its name in Alex's iron armour and change its ID. My ID is PovTutAlexArmour1. You could also check the box of Quest Item, which makes sure the player can't drop the item.

Now add a new container, a chest. You can change its name, but if its name is Chest already, you can just keep it that way. Or you could change it into Alex's chest, whatever you want. My ID is PovTutAlexChest1. Now, place the iron armour in the chest, and go to the lock tab of the chest. There you change the lock to "needs a key", and choose the key you've created a minute ago as the key which can unlock it. Now, you place the chest in a cave, or wherever you want. Then click on the chest, tick the box 'Persistent Reference' and add a reference ID. Mine is PovTutArmourChestRef

Quest data:

Phew... That was the item creating part. Now we are going to the part you came for in the first place, the quest itself! Click on the **Q** on the toolbar to open the quest window. At the left side you can find all the quests listed.)

When you are ready, R-Click on one of the quests in the Editor ID box, and click on "New" A small window will open, where you can enter your quest ID. Again, make it unique. Mine is PovTutAlexArmourQuest1

Now, select the ID you just created. Enter your quest name, mine is Alex's Armour. Change the priority to something high, 50 is fine. At the quest conditions, search for GetIsPlayableRace, and select it (don't press enter, it'll close the quest window).

Quest Stages:

That's all for the quest data tab, we will now go to the quest stages tab. At the Index, R-Click and select 'New', Set it to 10. The text you add to the Log Entry, is the text you will see when your journal updates. I wrote, "I met Alex, a man who had lost his armour. He asked me to find it. It is supposed to be in a locked chest in [write the name of the cell you placed the chest in here]. He gave me the key. I should try to find this chest, and give him his armour back" Again, don't press enter, it exits the quest window.

Now, make a new index again, but with the stage 15. This time, you say: "I found Alex his armour, I should bring it back to him". Now make one more index, with the number 20, and type "Alex was happy I brought his armour back, and rewarded me with 200 gold." Also tick the box "Complete quest" here. (Also remember to write StopQuest in the result script box, otherwise the quest is still running in the background and can slow down your machine)

Quest Topics:

Next thing is the topics. Go to the topics tab and R-Click on a blank spot in the Editor ID, and choose Add Topic. Then select GREETING and press OK.

R-Click with your mouse on one of the empty fields at the info box, and click on ‘new’. Type: "Hi, could you help me? I lost my armour, and I want it back." Add another topic by selecting add topic in the Editor ID, R-Clicking on a topic and clicking on 'new'. Give it a name. Mine is PovTutLostMyArmour.

To the right, there are a few empty fields, with the words add topic above them. ***Do not*** use this to add topics. There is a bug in the CS that makes multiple mods that add topics that way conflict, so they won't work. Instead, go to the result script and type: "AddTopic PovTutLostMyArmour" without the brackets.

What we just did? Well, when Alex greets you with his greeting, the topic PovTutLostMyArmour will be added to his topic list, so you can ask him about it.

Now, at the conditions part of the window, click on new. At the function select GetIsID. At the parameters, click on *Invalid*. It will give a long list. Remember the npc you made? Type the letters of his ID (that's PovTutAlex for me) and you will find it. Make sure you select the NPC, not the armour or anything like that. Click on OK. Now it will say

GetIsID PovTutAlex1 = = 1 (1 means yes, 0 means no). This means that when the NPC you talk to has the ID PovTutAlex1, he will select this greeting. If you don't do this, every NPC in the world will say this greeting!

Of course, we don't want him to say this when he has his armour back already! So select new again from the Conditions part of the window. This time select the function GetStage. Click on Invalid, and search for the ID of your quest. At the comparison, choose the symbol <. At the value, type 10. This means that Alex will only say this if the quest stage is smaller than (<) 10!

Add a new greeting with the text "Please find my armour". Go to the conditions and do the same as last time with GetIsID. For GetStage, choose the same quest, but select the comparison = =, and choose the value 10. So when he has asked for you to find the armour, but you haven't found it he will say this. (= = means, "equal to" So it does this when it's 10, not higher or lower.)

Now, add one more greeting with the text "Thanks again for finding my armour". Same GetIsID again. For GetStage, select = = 20. Don't forget the quest ID of course!

Now, at the editor's ID of the topics (where the topic GREETINGS is) right click, and select add topic. Choose the topic you created at the first greeting, for me that was PovTutLostMyArmour. Change the topic text to "Lost my armour", because you don't want the player to see PovTutLostMyArmour as the topic, right? At info, right click and select new.

Now you will enter the text that will be shown when you click on this topic. Type something like "I've lost my armour when I was hunting in [write the name of the cell you placed the chest in here]. I placed it in a chest and locked it, to see if any bandits came to steal it. But then I was attacked, and ran back here without taking the armour! Could you please get it from me? Here is the key for the chest."

If your text is too long it might not fit into the box. Just press ok (just make sure you did finish the sentence), right click with your mouse at an empty field in the response text and type the rest of the text.

At the result script, type "SetStage PovTutAlexArmourQuest 10" without the brackets. Then press enter and type Player.AddItem PovTutAlexKey1 1

The ID of the character who gets the item is first (for the player, that's just player) then comes a dot, then AddItem. Then a space, then the ID of the item you want to give, in this case a key, and then how many you want to give to the player. So when this text is said, you will get a journal update at stage 10, which says: "I met Alex, a man who has lost his armour…" And the rest of the text, and the key is added to the PCs inventory.

At the conditions box, once again choose GetIsID with the same ID as with the greetings. That is Alex's ID!

Now, right click at an empty spot in the info field and select new. Type: "I hope you will find my armour" Choose the same ID as with the greetings, and with getstage do = = 10. So this text will appear when you click on the topic lost my armour, Alex already told you he lost his armour but you haven't got the armour back yet.

We are almost there… Right click and select new at info again, and type "Wow! You found it, thanks! Here is your reward!"

Same GetIsID, at GetStage do = = 15.

At the result script, type SetStage PovTutAlexArmourQuest 20 and type player.additem gold001 200. Gold001 is the ID for gold, by typing 200 behind it you give 200 gold! And of course type Player.RemoveItem PovTutAlexArmour1 1 so you give him his armour back!

Add another line to the info field and type "It's not lost anymore!"
Same GetIsID, for GetStage do = = 20.
This is the text that will be shown if you ask about the lost armour after you gave it back. Exit the quest window by pressing ok.

The script:

Now click on the pencil, which will open up the script window.

Click on script, and select new.

ScriptName PovTutAlexArmourScript; *this is the ID of the script; this should always be on top of your scripts.*

Begin OnAdd; *when the item this script is placed on (we will do that next) is added to the PCs inventory, the following thing will happen.*

If (GetStage PovTutAlexArmourQuest == 10); *If the quest stage is at 10*
SetStage PovTutAlexArmourQuest 15; Set it to 15.
Endif; *Just a standard thing you have to type when you used "if"*

Endif

End

Press save and exit the window.

If for some reason your script won't work (gives errors when you try to save it), remove all the text behind the ; symbol. It should work though don't press enter in the middle of a script line either. Now, find Alex his armour (in the object window, not in the chest). At scripts it should say NONE. Scroll through it until you find your script. Select it and press OK.

At this point, if you would start playing the quest it will work. But I will show one more thing to make the quest easier: Quest markers.

Quest Markers:

Open the quest window. And click on quest targets. At target ref, right click with your mouse and select new. At the conditions add getstage PovTutAlexArmourQuest == 10.

At the quest target data, select the cell you placed the chest with Alex's armour in. For ref, select the ref of the chest you made. Mine was PovTutArmourChestRef If it isn't in the list, go to the chest in the cell, select it and see if the box "persistent reference" is ticked, and if you made a reference ID. If not, do so. Now, when the quest's stage is at 10, the marker will be placed on the chest, so the player can find it.

Make a new line at the quest target ref. At the conditions, do getstage PovTutAlexArmourQuest = = 15. At the reference, scroll to the cell Alex is in, and select his reference. When you take the armour now, and the journal updates to stage 15, the marker will be placed on Alex, so you can find him.

The End!

That's it. You've just created a quest. Remember that everything you've made won't appear in game unless you select your mod at the data files when loading Oblivion, so do that first before testing! To start the quest just talk to the NPC you placed.

Maybe you could try to make another quest, and don't look at the tutorial too much. It's a good way to learn. And I'll say this again: SAVE OFTEN! If you mod for an hour without saving and the construction set crashes, you're likely to get frustrated.

My First Script, Scripting Tutorial:

Refer to The Scripting Window for more information on scripting.

Before we really start writing our tutorial script we should decide what we want it to do! For this tutorial we are going to make a *Riddle Cupboard*: The cupboard will ask a riddle and only the right answer will open the cupboard. If the player provides the wrong answer, a trap will go off, hurting the player, and the cupboard can't be opened. That's a fairly complex undertaking, but we will take it step by step and see that it's not so bad after all.

Writing the script:

Once you have the Script Edit window open, click *Script -> New*.. You should see that the *Script Type* is "Object", which is exactly what we want it to be. Click in the main part of the window, which should now have gone from grayed out to white. This is where you'll be writing the script.

Naming the Script:

First of all we must give our script a name. Every script must start with a declaration of that script's name. In the editing field, please type:

```
ScriptName RiddleChestScript
```

Note that there are no spaces and no underscores in the name. Your script name must be one word, so you may not use spaces. Also, Oblivion ignores underscores in the alphabetical listing of scripts; you may use underscores to make the title clearer for yourself, but you will not see them when you look to open the script later. (Advanced scripting information: Oblivion allows something called *aliasing* in TES Script. This is where you replace the full name of the command with a shorter version. For instance, "ScriptName" becomes "scn", or "ForceActorValue" becomes "ForceAV". We will not be using aliasing in this tutorial, but don't be surprised if you see other scripts that do use it.) Remember that TES Script is not case sensitive; you could have written "scriptname" or "ScRiPtNaMe" just as well.

Try to save your script using the *Save* button in the toolbar. Nothing obvious should happen, but if you click on the *Open* button, you should see your script name in the list of scripts in Oblivion. Click on the X at the top right corner of the *Select Form* window to close it and return to the Script Edit window. RiddleChestScript is now the shortest script possible in Oblivion, but it doesn't do anything right now. Let's change that.

Begin and End:

The Begin and End commands define, somewhat obviously, the beginning and end of blocks of the script—separate chunks of code that will be performed under certain conditions. For example, "Begin GameMode" defines the start of a block of text that will run every frame the user is not looking at a menu. For our purposes, we want an OnActivate block: one that will run when our chest is activated, but no other times. (It would get very annoying having to answer the riddle every frame, or every time an NPC died, both of which are possible using other variants of Begin!) So hit enter twice to move the cursor down, and edit your script to look like this:

```
ScriptName RiddleChestScript

Begin OnActivate
   ; Here we will enter what happens when the chest is opened.
End
```

Click *Save* again now. The script still doesn't do anything, but at least now we have defined under what conditions it will run. There are a couple of things to note here. First, we have ended the current block before beginning a new one—not so important when there's only one block, but as this script gets more complicated we'll need multiple blocks in a single script. In somewhat more technical terms, Begin/End blocks do not nest. The other thing to notice is the semicolon ";" on line 4. A semicolon marks the rest of the line as *comment*. Whatever you type after the semicolon *on that one line* will be ignored when the script compiles. You use comments to explain your code, which can help you and others understand what's going on inside the script quickly at a later date. If you want more than one line of comment, you must have one semicolon for each line.

MessageBox Function:

Now we want our trapped chest to ask the player a riddle. For this we use the *MessageBox* function, which should look familiar: the My First Script tutorial used a close relative of MessageBox, the Message command. MessageBox allows us to display some text on the screen and also to display choices that the player can select from. Unfortunately Oblivion (like Morrowind before it) has no option to have the player type in the answer to our riddle, so we will have to give multiple choices. The line for that could read:

```
MessageBox "Voiceless it cries, wingless flutters, toothless bites, mouthless
mutters. What is it?", "Bat", "Old woman", "Wind", "Wraith"
```

The first *string* (the text between the first pair of quotation marks) is the text actually displayed in the box; the other texts, separated by commas, tell the game to make "buttons" with the given text displayed one string per choice.

But how do we ensure that the riddle is asked only the first time we try to open the chest and not every time? We now come to a very central point: the use of do-once conditions and state variables. Most of the problems that beginners encounter with scripting for Oblivion have their roots in misunderstanding how the scripts are actually executed and how scripts should accordingly be structured. So let's have a look at this in greater detail. Remember to save regularly.

How Object scripts are executed:

Every script that is attached to an object or an NPC (Object script) is executed every frame the game displays on screen while the cell with the object is active (when indoors only the cell the PC is currently in is active, when outdoors the PC's cell and all adjacent cells are active). So the complete script (not just one line of it) is executed 10-60 times a second or however fast your computer runs the game! It is best to imagine every local script wrapped in a big "whileloop":

```
while (Object is in active Cell)
  [Your script code]
endwhile
```

This is the reason why the following script spits out a continuous stream of messages (if attached to an object or NPC in the same cell as the player). Try it, if you want:

```
ScriptName HorribleMessageScript

Begin GameMode
  MessageBox “Thousands of useless messages!”
End
```

This example is relatively harmless, but imagine what happens if you had used a line of code that adds an item to the player's inventory, or places a monster next to him, etc.!

For this reason, “Do Once” constructions are very essential and something you will probably use a lot while scripting for Oblivion. So, let's go on with our tutorial script: we need to declare a variable and use it to make sure the message is only displayed once. Change the script to the following:

```
ScriptName RiddleChestScript

Short controlvar

Begin OnActivate
  If ( controlvar == 0 )
    MessageBox "Voiceless it cries, wingless flutters, toothless bites, mouthless
mutters. What is it?", "Bat", "Old woman", "Wind", "Wraith"
    Set controlvar to 1
  EndIf
End
```

Save the script again. Note that the MessageBox command should be on one line in your script! Do not break it up!

Once again, there are a few things to point out in this latest addition. The "If" command is there to check a condition—whenever the expression in the parantheses is true the following lines of code will be executed until the "EndIf" command is encountered. The "==" checks if an expression on the left of it (in our case the variable controlvar) is equal to the expression on the right of it (in our case to 0). If you forget the EndIf command after an If command, the editor will complain with an error message when you try to save.

"Short controlvar" declares a new variable we'll call "controlvar", of type short. For the moment it's enough to know that this is variable that will contain integers (whole positive or negative numbers). A variable is a "placeholder" that can take on different values. The Set command is also new, but simple enough—it sets our variable that had the value 0 before (all variables start out at zero when declared) to 1. This, in connection with the "If (controlvar == 0)" command, provides a do-once condition—the next frame the script is executed after the variable was set to 1, the If condition will be false and the message box will not be displayed again.

Saving and preparing the mod:

Now our script is already capable of being run, so lets test it:

1. Save the script and close the script editor window.

2. Look to the Object Window, and proceed in order through the directories of *WorldObjects* > *Container* > *Clutter* until you find the object with the Editor ID of CupboardFoodLower.

3. Now either right-click and choose Edit or double-click the object to bring up its properties.

4. In the script dropdown menu, select the script you just made RiddleChestScript.

5. Hit OK, save the mod, and quit TESCS.

Note: *Pay attention to what we have just done. Editing an object without giving it a new ID is terrible modding practiceNever do this unless you are sure what you are doing!*

The script in-game:

1. Now use Data Files to activate your mod, start Oblivion, and load a savegame.

2. When your game has loaded, bring down the console (usually the ~ key, or whatever you have to the left of the "1" on the main keyboard) and in the console type:

```
player.coc "AleswellInn"
```

And press enter.

- You will appear in the Aleswell Inn, and be immediately greeted by a group of invisible people. This has nothing to do with the mod, but relates instead to the author's random choice to use this cell to test the mod (it's near the top of the list of interiors, so is very easy to find). Click through until you can move, and approach the cupboard on the wall to your left.

When you activate the cupboard, you should see your messagebox appear on the screen.

Clicking on any of the choices should close the messagebox for the moment, and nothing should happen when you activate the cupboard again—which is good, because it means our controlvar check is working like we wanted. (The cupboard is not activating because we have not included an Activate command anywhere in our script, but we'll come to that later.)

Quit Oblivion and reload your mod in the CS. Those poor people in Aleswell Inn will have to wait till some other time to regain their visibility, but such is life being an NPC in a video game.

Letting the player choose an answer:

We now need to figure out which answer the player selects, and script appropriate reactions for right and wrong answers. The function to test the selected answer is GetButtonPressed. This function returns a number depending on which of the buttons of a message box has been clicked on with the mouse. It will return "0" for the first button ("Bat" in our example) and 1, 2, 3, etc. for the following buttons, in the order you listed them in the MessageBox command. Until an answer has been selected, the function will return –1, so we have to take care of that, too.

The "Activate" function will make our cupboard open; in fact, Activate will simply trigger the standard action that would usually be performed when you "use" the scripted object—doors will swing open, NPCs will initiate dialogue, etc. The following update to our script also demonstrates how you can use a control variable to force Oblivion to process functions one after the other, although the complete script is processed every frame of the game: simply increment the control variable and test it in a series of If–ElseIf statements. This is a very safe way of scripting for Oblivion. It may not always be necessary, but it's safe.

Please edit the script to look like the following:

```
ScriptName RiddleChestScript

Short controlvar
Short button

Begin OnActivate
  If ( controlvar == 0 )
    MessageBox "Voiceless it cries, wingless flutters, toothless bites, mouthless
mutters. What is it?", "Bat", "Old woman", "Wind", "Wraith"
    Set controlvar to 1
  ElseIf ( controlvar > 1 )
    Activate
  EndIf
End

Begin GameMode
  If ( controlvar == 1 )
    Set button to GetButtonPressed
    If ( button == -1 )
      Return
    ElseIf ( button == 2)
      MessageBox "Your answer was correct."
      Activate
      Set controlvar to 2
    Else
      MessageBox "Your answer was wrong."
      Set controlvar to -1
    EndIf
  EndIf
End
```

Take a look at the new GameMode block that starts with "If (controlvar == 1)". We have set controlvar to 1 as soon as the cupboard got activated. But OnActivate only runs script for the frame the object was activated; we must use the GameMode block to continue checking for the player's choice, since that runs every frame. Now we test for which button is being pressed. We do this by assigning the new variable "button" a value returned by GetButtonPressed. Since the script is still running, even while the game seemingly pauses to await your decision, we first test if no button has been selected yet—return tells the game engine to stop processing the script for this frame.

Our correct answer was "Wind", which corresponds to button number two—if button number two gets pressed, we want to tell the player that he gave the right answer, and allow Activate to open the cupboard's inventory in the usual way. All other values of button mean that the player has selected a wrong answer, so we can use the Else command here. In this case we tell the player what a fool he was and the chest is not activated.

Now look at the little addition at the top of the script:

```
Begin OnActivate
 If ( controlvar == 0 )
  MessageBox "Voiceless it cries, wingless flutters, toothless bites, mouthless
mutters. What is it?", "Bat", "Old woman", "Wind", "Wraith"
  Set controlvar to 1
 ElseIf ( controlvar > 1 )
  Activate
 EndIf
End
```

This means that, whenever the cupboard is activated in the future, it will only open if controlvar is greater than 1. Remember the last paragraph: if the player provides the wrong answer to the riddle, controlvar is set to –1, so he will never be able to open the cupboard. But if he knows the right answer, controlvar is set to 2, and from now on the player can open the chest as often as he likes. 'Save and run your plugin, and test as described above.

Your first bugs, and the fixes:

If you tested the plugin, you should already have noticed that things do not go quite as planned. Choosing incorrectly does lock the chest forever, but if the player chooses correctly the inventory and the correct-answer messagebox come up at the same time. The messagebox must be cleared before anything else may be done, but it's covered by the inventory!

Let's try the following (change the corresponding section of your script to look like this:

```
 If ( controlvar == 1 )
  Set button to GetButtonPressed
  If ( button == -1 )
   Return
  ElseIf ( button == 2)
   MessageBox "Your answer was correct."
   Set controlvar to 2
  Else
   MessageBox "Your answer was wrong."
   Set controlvar to -1
  EndIf
 ElseIf ( controlvar == 2 )
  Activate
 EndIf
```

See how we moved the activate command to the section that tests for controlvar == 2? This provides a cleaner sequence of events by preventing the two boxes from being open simultaneously, and as was mentioned above, clean sequences of events can be very important when scripting for Oblivion—always try to avoid doing too many things at once! Well, save, run and test it.

Great, now the inventory opens as we wanted, but what is this? We can't close the inventory! Look above:controlvar was set to two, and remains there since we do not change it again. Therefore the game now gets continuous "Activate" commands each time the script is processed (every frame)! That's why we can't close the inventory—it gets reopened immediately. So change the following part of the script:

```
 ElseIf ( controlvar == 2 )
   Activate
   Set controlvar to 3
 EndIf
```

Test the mod again: now everything works the way we wanted. Hopefully you are not confused from with the above excursion into the process of debugging, but it is a very important thing to know about—you will constantly have to rethink your scripts and try different ways of doing it to be successful.

What's missing now? The trap effect, of course!

Adding the trap:

Our cupboard will put a curse on the player if he fails to answer the riddle. First, select the spell you want to hit the player with: click on the + sign next to Magic, the + sign next to Spell, and click to highlight Spell. There are quite a few choices here, but for the purposes of the tutorial we'll use the Mg05FingerSpell15 spell.

With our painful consequence chosen, we need to add it to the player when he chooses the wrong answer. Edit the script again:

```
   Else
     MessageBox "Your answer was wrong."
     Cast Mg05FingerSpell15 Player
     Set controlvar to -1
   EndIf
```

Note the use of the Cast function. (Advanced scripting information: Cast requires a calling object to work. We did not include one in our script because it is an Object script, and will be attached to an object in the game. The script will therefore default to the attached object when functions require a calling object.)

Now, your script should look like this:

```
ScriptName RiddleChestScript
```

```
Short controlvar
Short button

Begin OnActivate
  If ( controlvar == 0 )
    MessageBox "Voiceless it cries, wingless flutters, toothless bites, mouthless
mutters. What is it?", "Bat", "Old woman", "Wind", "Wraith"
    Set controlvar to 1
  ElseIf ( controlvar > 1 )
    Activate
  EndIf
End

Begin GameMode
  If ( controlvar == 1 )
    Set button to GetButtonPressed
    If ( button == -1 )
      Return
    ElseIf ( button == 2)
      MessageBox "Your answer was correct."
      Set controlvar to 2
    Else
      MessageBox "Your answer was wrong."
      Cast Mg05FingerSpell15 Player
      Set controlvar to -1
    EndIf
  ElseIf ( controlvar == 2 )
    Activate
    Set controlvar to 3
  EndIf
End
```

Ok, now we have our final working script. Congratulations! If you want, experiment a little more with this script using some of the other functions in TES Script.

Making your first NPC Tutorial:

In this tutorial I will be guiding you through making your first NPC. I recommend reading the NPC Window before here, as I will not be giving a whole lot of detail.

Make sure you have a new NPC window up.

A rundown of the NPC:

Before I begin creating an NPC I like to write down what the NPC is going to be named and some other general details. This NPC is fairly basic; here is the rundown of the NPC we will be creating:

Name: Tracey

Sex: Female

Race: Imperial

Class: Warrior

Description: Tracey is a young woman in her early twenties; she wears a full suit of steel armor besides the helmet; and wields an Iron Battleaxe.

As I said, this is a very basic NPC. If you were making a large mod you would write down how she looks, what her schedule is and various other things.

Creating the basics of the NPC:

Ok, first give your NPC an ID; make sure that it's unique. (If I were to make an ID I would type: LGDTracey Why? My Online Name is Lord_Gannondorf. The L stands for Lord, the G for Gannon and the D for Dorf. And Tracey is the NPCs name.) Doing this will make you mod have less chance conflicting with another mod making the two unplayable.

Just below the ID box is the Name box, put the NPCs name here, Tracey.

Next we want to give Tracey a class, we want her to be a warrior; so in the Class drop down box find Warrior and select it.

A bit further down we will see Level, give her a level of 7, and to the right of the window check the box that says '*Auto Calc Stats'*. As you see most of the stats tab is grayed out and her stats have been automatically calculated.

Next change her race to Imperial, and check the box that says 'Female'.

Filling the NPCs Inventory:

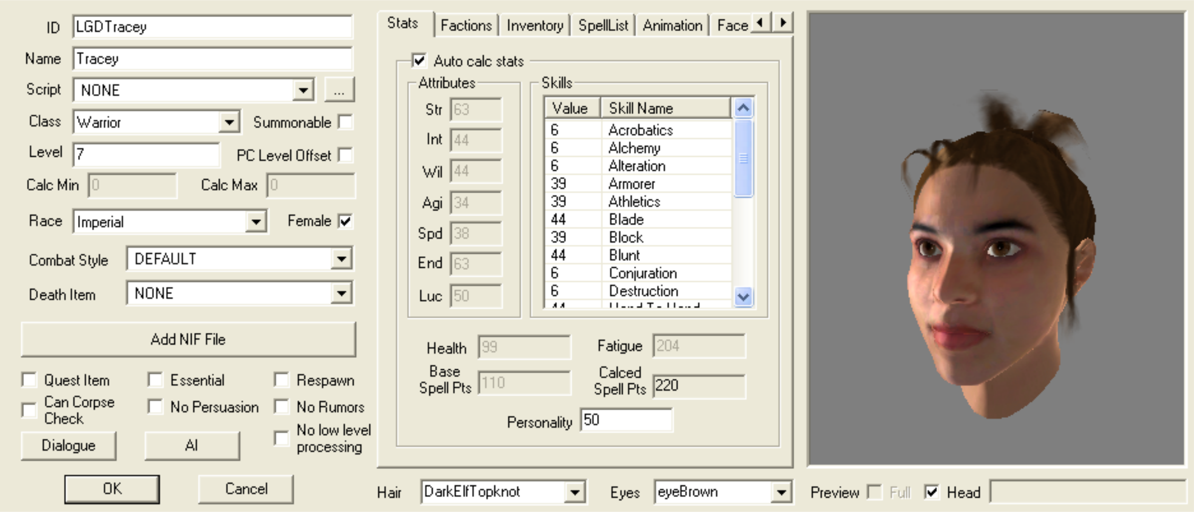
Now that we have done the basics we need something for her to wear! At the top of the window click on the *'Inventory'* tab. Now, keep the NPC window up and navigate through the **Object Window** until you have found armor. Look through the armor list and click on steel. Then drag and drop SteelBoots, SteelCuirass, SteelGreaves and SteelGauntlets from the **Object Window** to the NPCs inventory. Navigate through the menus again and find WeapIronBattleAxe under weapons and drag it into her inventory. She will now have the armor and weapon in game.

Finishing of the NPC:

Ok, so now we just have to finish off Tracey. Click on the *'Face'* tab; to the very bottom of the window you will see the drop down menu that let's you select her hair, select any hair you wish.

Move around the age slider until she looks about 20 years old, then move the complexion slider to make her look even better. Mess around with this page and then read the next paragraph.

We are almost finished, next click on the 'Face Advanced' Tab, you will see a list of selections to the left and a slider beside them. Click on 'Brow Ridge – high/low', now move the slider up and down and you will see her eyebrows go up and down; neat eh? Mess around with these settings until you are satisfied. Your window should now look something like this:



ID LGDTracey
Name Tracey
Script NONE
Class Warrior
Summonable ☐
Level 7
PC Level Offset ☐
Calc Min 0
Calc Max 0
Race Imperial
Female ☑
Combat Style DEFAULT
Death Item NONE

☐ Quest Item
☐ Essential
☐ Respawn
☐ Can Corpse Check
☐ No Persuasion
☐ No Rumors
☐ No low level processing

☑ Auto calc stats

Str 63
Int 44
Wil 44
Agi 34
Spd 38
End 63
Luc 50

Value	Skill Name
6	Acrobatics
6	Alchemy
6	Alteration
39	Armorer
39	Athletics
44	Blade
39	Block
44	Blunt
6	Conjuration
6	Destruction

Health 99
Fatigue 204
Base Spell Pts 110
Calced Spell Pts 220
Personality 50

Hair DarkElfTopknot
Eyes eyeBrown
☑ Head

The End!

There are a lot more features of NPC editing than we explored here! Have a mess around and see what you can figure out, hope you enjoyed the tutorial!

Enchanting your First Weapon Tutorial:

In this tutorial you will learn to put a simple enchantment on a steel Longsword, the enchantment will be frost damage. So, lets begin:

Getting the Weapon Ready:

Go to Items->Weapon->Weapons->Steel and find WeapSteelLongsword and D-Click on it. Change ID of the weapon to something unique. (If I were to make an ID I would type: LGDFrostBite Why? My Online Name is Lord_Gannondorf. The L stands for Lord, the G for Gannon and the D for Dorf. And Frost Bite is the weapons name.). Make sure you also change the name of the weapon to Frost Bite.

Now we have done that click 'Ok', a message will appear asking if you want to create a new form. Press yes to create a new item.

Creating the Enchantment:

Now go to Magic->Enchantment and R-Click in the list and press new, give the enchantment an ID; mine is LGDFrostBiteEnch (LGD – Lord_Gannondorf, FrostBite – Weapon Name, Ench – Enchantment.)

Make sure the 'type' drop down box is set to Weapon otherwise the enchantment won't work. R-Click in the effects list to the right and press new. The Effect Item window will pop up; change the effect drop down menu to Frost Damage. Next change the magnitude of the effect to 15; the magnitude is how powerful the Frost Damage is. Press 'Ok', then 'Ok'.

Giving the weapon the enchantment:

Find the weapon that we created in the Object Window and D-Click it. In the Enchanting drop down box find the enchantment that we just created, just below the enchanting drop down box you will see Enchantment. Put 1200 in here; this is the enchantment power of the weapon.

The End!

Now just place the weapon in the game and you are ready to go! Hope you enjoyed the tutorial.

Making a House Tutorial:

In this tutorial I will try to teach you how to make a house in the CS.

Ok, first we need to decide what house design is going to be, what sort of furniture will be in the house and what sort of miscellaneous objects (food etc.) will be in the house.

This is a basic layout of the house:

> The house is a low-end design (waterfront), which doesn't contain a basement. There needs to be at least one bed for the player to rest in, preferably a single-bed, the house will be designed for a warrior character, there will be weapon stands, hay targets, etc. The objects will not be placed neatly around the house; they will be 'thrown' around and the house will be messy. The house is located just outside of the Imperial City in the small farm town 'Weye'.

Now that we know what we want in the house we should start building it! Let's start by creating the interior cell for the house. To find the Cells Window go to World->Cells.

Once the window has opened R-Click in the list to the left and click new; give the house an ID, for this tutorial we will make the ID: 'MyFirstHouse'.

The Common Data tab is fine so let's move on to the lighting tab. Here is where you can adjust lighting in the cell, add fog etc. In *'Ambient'* Give R, G and B the following values:

R: 35
G: 40
B: 50

This determines the lighting for the cell; if you keep all these values at 0 the room would be black; but we still need to add candles etc. Next go to the Interior Data tab, in the name text box type: My Shack. This is the name of the cell in game.

Now we have the cell ready to place objects in. Find the cell in the Cell View Window (Not the Cells Window!) and D-Click it; this will load up a blank cell in the Render Window.

In the Object Window we need to find the interior for the house, go to WorldObjects->Static->Architecture->LowerClass->*HouseLowerInterior01* to find the house. Once you have found it drag and drop it into the render window. You will see the interior of the house sitting there, now we need to add some furniture.

In the Object Window find Static->Clutter->LowerClass->*LowerClassTable0.* Once again drag and drop the object in the render window. You will see the table appear in the render window, it may not be in the exact place you want it to be; to move it around hold L-Click on the object and move your mouse. But, how do you move the table up and down? All you need to do is hold 'Z' on the keyboard. Look at Shortkeys for more information.

Now that we have the table there we need a chair to sit on, go to WorldObjects->Furniture->LowerClass in the object window. Here is where all the beds, chairs etc are. If you drag and drop a chair into the object window you will see what seems to be a 'ghost' sitting on the chair and one to the left, right, front or behind it. The ghost on the chair is where the player will be sitting and the other one is which way the player will get on the chair when it's activated. Choose a chair and sit it near the table, you can also add a bed in the house while you are adding the chair.

Now you know how to clutter your house with furniture, all you need to do know is add objects (food, cutlery etc) to the house. You will find all of these items under Items->MiscItem in the object window.

Once you have cluttered your house we need to make the house accessible from the outside. First we need to add a door to the interior, in the object window go to Static-

>Door->LowerClass->DoorFullLower and drag it into the render window. Make sure you position the door in the ‘door hole’.

Now for the outside, on the cell view window set the World Space drop down box to Tamriel, in the list find WeyeExterior and D-Click it. Find an empty space here somewhere and place the exterior of the house (WorldObjects->Static->Architecture->LowerClass->*HouseLower01*). Once you have placed it put another door in the door hole.

Now we need to ‘connect’ the doors so the player can get to the interior. D-Click on the exterior door and click the teleport tab, check the Teleport check box and change the Cell drop down box to MyFirstHouse and set the reference to DoorFullLower.

Now the doors are linked but we still need to position the teleport markers, click on View Teleport Marker and move the selected Teleport Marker to the correct place (The teleport marker controls where the player appears when the door is activated). Position both door makers correctly.

You are now ready to test your house in game, remember to activate the plug-in in the Data window in the Oblivion Launcher.

Chapter 8 – Tweaking Oblivion

Editing Oblivion.ini

By default, each time Oblivion loads up, it reads the values held in the *Oblivion.ini* file stored under your *\Documents and Settings\[username]\My Documents\My Games\Oblivion* directory. This file can be opened and edited using a text editor like Windows Notepad or WordPad, however before editing it, first change all of your in game settings to the way you want them, then close Oblivion and make a backup copy of this file. If at any time you want to find the default values for any of the .ini variables, open the *Oblivion_default.ini* file under your *\Program Files\Bethesda Softworks\Oblivion* directory. This file contains all the defaults for Oblivion on your system, and should never be edited or deleted. If you want the game to quickly restore all the default values for your Oblivion.ini file, simply delete the Oblivion.ini file and it will be recreated with the default values the next time you start Oblivion. This can help to quickly resolve any problems you may be having as a result of incorrectly tweaking variables, however obviously you will lose all your customisations as well as your in-game settings preferences.

The performance impact of these tweaks will vary from machine to machine, sometimes significantly, but the general impact is indicated. Note that I have tested *every* variable in the Oblivion.ini file, and those which seem to have no discernable or useful impact, or which can cause crashes or odd behaviour are not included below. Some of the variables people have been using do not do what they think they do, so if you see some notable exceptions, bear in mind that it is likely they are either not useful, or can be adjusted in-game - settings which can be fully altered using the in-game settings aren't shown below, unless they can be tweaked further.

Note: *If you're having problems finding any of these commands in your Oblivion.ini file, first make sure you're using the right .ini file (read the info above again carefully), then go to the top of Oblivion.ini, press CTRL+F to open the search box, type or paste the name of the command into there, and it will find it for you. All commands below should be in everyone's .ini file.*

[General Variables]

BAllowConsole=1 - This setting allows you to use the in game console if set to 1. There is no reason to set it to 0.

FDefaultFOV=75.0000 - Determines the default Field of View whenever you start Oblivion. If set differently from the default of 75 degrees, can put screen elements out of synch and also insert borders around the screen.

FGlobalTimeMultiplier=1.0000 - Raising the value above 1 speeds up the rate at which world time passes, lowering it below 1 slows down time. This setting is interesting for watching rapid-motion sunsets/sunrises for example.

BBorderRegionsEnabled=1 - If set to 0, this option removes the invisible barriers around the outer parts of the game world, though not much is beyond.

IMaxDecalsPerFrame=10 - This value determines the maximum number of decals (blood marks) visible on screen at any time. Note that decals can have a noticeable impact on performance; so raising this value can reduce FPS during combat.

FDecalLifetime=10.0000 - This value determines the time in seconds before decals (blood marks) disappear. The higher the value, the longer the blood marks will take to fade away. Note again that decals can have a noticeable impact on performance.

FMinBloodDamage=1.0000 - This value determines the minimum amount of damage you must inflict to draw blood.

BUse Joystick=0 - Set to 0 if you don't have a joystick connected, some people claim this improves FPS and resolves mouse/input lag.

BInstantLevelUp=0 - If set to 1 allows your character to instantly level up without having to rest in a bed.

BSaveOnInteriorExteriorSwitch=1 - Controls whether to autosave whenever you enter/exit buildings. Setting to 0 can reduce loading pauses during entry and exit of buildings.

BPrecipitation=1 - If set to 0, disables rain effects.

BCheckIDsOnInit=0 - This variable appears to be for troubleshooting purposes, in that it checks to make sure object/NPC IDs are correct as the game loads. There is no impact from setting this to 1, so it is best left at 0 unless you are having quest problems for example.

[Graphics Variables]

BFull Screen=1 - Determines whether to start Oblivion in Fullscreen/Windowed mode -. Useful for changing here in case you have problems with the Oblivion Launcher.

The below two values determine the screen width and height in pixels respectively - you can specify a custom screen resolution (in Windowed mode only) by changing these values.

ISize W=1280

ISize H=1024

The below settings are useful particularly for those with older monitors - you can alter the limits on the gamma (Brightness slider), and hence allow further adjustment of the brightness slider to suit your monitor.

FGammaMax=0.6000

FGammaMin=1.4000

IShadowMapResolution=1024 - This variable determines the resolution of the shadow maps. By lowering this value (always using a multiple of 8), you can gain significant FPS in areas with dynamic shadows in return for much 'rougher' looking shadows. For example, try a value of 128 to see the performance and image quality impact.

BAllow30Shaders=0 - If set to 1, this option allows (but does not force) the use of Shader Model 3.0 on graphics cards which support it, namely Nvidia GeForce 6600 or newer, or ATI X1000 series or newer. This can potentially improve performance when using HDR rendering for example. Check your *RendererInfo.txt* file in your *\Documents and Settings\User\Documents\My Games\Oblivion* directory to see if your card supports SM3.0 next to the option '3.0 Shaders'. Note however that even by enabling this option, Oblivion still appears to use 2.0 shaders (check the 'PSTarget' and 'VSTarget' lines in Rendererinfo.txt). In any case, if you have one of the cards mentioned above, it cannot hurt to enable this option.

The below two options allow you to raise the maximum number of Interior and Exterior Shadows possible (as set by your in-game sliders). The normal maximum is 10, and obviously raising this can reduce FPS.

IActorShadowIntMax=10

IActorShadowExtMax=10

FSpecualrStartMax=1000.0000 - This option determines the maximum range of specular lighting, if enabled. Lowering this value may provide some additional FPS in outdoor areas, without having to turn off specular altogether. Alternatively you can raise it further to gain specular lighting on distant objects. Note that the word specular has been misspelled as 'specualr' in the name of this variable - don't correct it, as the misspelled version is the one the engine detects.

FShadowFadeTime=1.0000 - Determines how many seconds it takes for shadows to fade in/out as you approach/retreat from objects/characters which cast shadows.

BAllowPartialPrecision=1 - This setting determines whether the shaders run in Partial Precision DX9 mode. Essentially this should be kept at 1, since partial precision provides the best FPS for minimal or no noticeable image quality loss, particularly on Nvidia FX graphics cards. However if you want the best possible image quality, set this to 0 at the cost of some FPS.

BUseRefractionShader=1 - This setting controls the shimmery/invisibility effect. Setting it to 0 can noticeably boost FPS in areas where this effect is used, such as around invisible characters, or at an Oblivion gate. Note in particular that ATI users who have major problems in such areas in particular should set this to 0, though that in turn may cause other problems.

BDoTexturePass=1 - Setting this to 0 removes textures from most objects. It can improve FPS but is obviously not recommended.

BDoSpecularPass=1 - If set to 0, removes the shiny effect on most appropriate surfaces This can noticeably improve FPS on many systems, but can also cause crashes in certain areas (e.g. Weynon Priory). You can try to reduce these crashes using this mod.

BDoDiffusePass=1 - If set to 0, removes all dynamic lighting and hence is not recommended.

BDoCanopyShadowPass=1 - If set to 0, removes all tree shadows, which can improve FPS in forested outdoor areas.

BLocalMapShader=1 - If set to 0, removes the brown hazy overlay on the local area world map, making the colours much brighter but also highlighting the relatively low resolution of the map.

BFullBrightLighting=0 - If set to 1, alters the global lighting method to one which is far less detailed and doesn't use complex shaders, hence is much less strenuous for older cards. The results are graphically unpleasant as most textures look terrible, and there will be some graphical glitches, but this should allow older cards to run Oblivion more smoothly.

MaxLandscapeTextures=0 - If set to 1, this setting appears to increase the use of generic landscape texture in place of more specific ones. This saves on texture memory, but can cause graphical oddities, such as roads being replaced with grass textures, or clear lines where different types of textures should blend into each other. It may also cause problems with the LOD Texture Replacement mods, and hence is only recommended for those with very low-end graphics cards struggling with stuttering and low FPS.

BLandscapeBlend=1 - If set to the default of 1, it provides smoother blending of distant LOD textures with closer detail textures as you approach objects/terrain. This setting should be left enabled, as it prevents the previously unrealistic way in which distant textures went from being very blurry to suddenly becoming sharp and distinct as you approached them. It should also result in less dramatic loading pauses as you move around.

[Audio Variables]

BDSoundHWAcceleration=1 - If set to 1, uses hardware acceleration (i.e. your sound card) to reproduce audio. This provides the best audio quality, however if you are having problems such as odd crashes, you can try a value of 0 to disable hardware sound acceleration.

BMusicEnabled=1 - If set to 0, turns off all background music. This will detract from the atmosphere of the game, but can noticeably improve stuttering on some systems, as background music dynamically loads throughout the game.

BSoundEnabled=1 - If set to 0, turns off all sound effects, but does not affect music. This would be an extreme way of gaining performance or reducing crashes and is not recommended for anything other than troubleshooting purposes.

FMainMenuMusicVolume=0.6000 - The main menu music volume can't be adjusted anywhere in the game, except by using the Master Volume slider (which then affects all sound/music). To alter the menu music volume independently, change the value here.

IMaxImpactSoundCount=32 - This option determines the maximum number of channels used for different sound effects. Reduce it 24 or 16 to remove some of the sound effects in return for a performance boost and stuttering reduction. Note however that this setting can crash your system if set too low, particularly with hardware acceleration enabled.

[Memory, Loading & Multithreading Variables]

All users should implement the Memory and Loading tweaks (with appropriate values) below, but the Multithreading tweaks are best used on Dual Core or HyperThreading CPUs.

Memory Tweaks:

UInterior Cell Buffer=3

UExterior Cell Buffer=36

The above values determine how many cells (whether for interior or exterior areas) are buffered into RAM. Note that usually the value of the Exterior Cell Buffer variable is automatically set by the engine based on the size of the *uGridstoLoad* variable. The higher that variable is, the higher the engine will raise this value. However if you want to smooth out your FPS, try manually setting a higher value for both of these, depending on how much RAM you have. For 1GB I recommend doubling the values (6 and 72 respectively). For 2GB of RAM, I use 16 and 102 respectively. For higher amounts of RAM, try raising them higher, however note that you should also raise the *iPreloadSizeLimit* value below.

IPreloadSizeLimit=26214400 - This setting appears to determine the maximum amount (in bytes) of RAM allowed for preloading game data. The higher the value, the more chance you have of reducing stuttering. The default value equates to around 25MB (divide the setting by 1024 to get KB, then by 1024 again to get MB). For those with 1GB of system RAM, try doubling the variable to 52428800. For those with 2GB, try double again at 104857600 (100MB). You can raise these values even further to experiment, however note that raising this to a large amount doesn't force all the game data to sit in RAM, and can actually cause crashes. I suggest the maximum anyone should set this to should be around 262144000 (250MB), even for 2GB of RAM. Make sure to raise your Cell Buffer values accordingly (see above).

BPreemptivelyUnloadCells=0 - If set to 1, this setting attempts to unload cell data it thinks you won't need. This can help those with less than 1GB of RAM, however with 1GB or more of RAM, I recommend leaving it at 0 for greatly reduced stuttering.

Note: *As you raise the values of iPreloadSizeLimit and the Cell Buffer variables further above, you should make sure bPreemptivelyUnloadCells is set to 0 to take advantage of such additional memory allocation, otherwise it will actively work against the benefits brought about by the other tweaks.*

BSelectivePurgeUnusedOnFastTravel=0 - If set to 1, this option removes a range of unnecessary data when you Fast Travel to another location. This can help keep memory usage down for those with less RAM, so it is recommended such people set this to 1, otherwise leave it at 0.

Loading Tweaks:

BUseHardDriveCache=1 - Although Windows should be using your hard drive cache by default for all drive operations, setting this variable to 1 should ensure it does. Note that some people report that enabling this setting can increase stuttering, so experiment to see if it helps or makes things worse for you. If in doubt, leave it at 0.

BBackgroundLoadLipFiles=1

BLoadBackgroundFaceGen=1

BBackgroundCellLoads=1

BLoadHelmetsInBackground=1

IBackgroundLoadLoading=1

BBackgroundPathing=1

All of the above options relate to background loading to attempt to smooth FPS and reduce stutter. I recommend setting them all to 1. This may increase some transitional loading times (e.g. loading cities, crossing indoors/outdoors), but should generally reduce random loading stutter as you wander around.

BUseBackgroundFileLoader=0 - This setting may improve stuttering performance on some machines if set to 1, but I have noticed that it can also cause crashes and long loading times for some people, so on balance I recommend leaving it at 0.

IPostProcessMillisecondsEditor=50

IPostProcessMillisecondsLoadingQueuedPriority=20

IPostProcessMilliseconds=5

I'm not certain of what the above three settings do, but they seem to relate to the loading of post process effects such as HDR which are applied after a scene is rendered. By changing these values you can fine-tune the way their loading impacts on stuttering in the game world. I tried various values, but the most noticeable impact was reducing the *iPostProcessMillisecondsLoadingQueuedPriority* value from its default of 20 to a lower value, such as 5. It will depend on your particular system as to whether changing this will have the same impact.

Multithreading Tweaks:

BUseThreadedBlood=1

BUseThreadedMorpher=1

BUseThreadedTempEffects=1

BUseThreadedParticleSystem=1

BUseMultiThreadedTrees=1

BUseMultiThreadedFaceGen=1

INumHavokThreads=5

IThreads=9

IOpenMPLevel=10

All of the above settings seem to relate to the use of the GameBryo engine's multithreading capability. Multithreading splits tasks into 'threads' where possible, and runs them in parallel across both cores of Dual Core or HyperThreading (virtual dual core) CPUs to improve performance. Note that raising the values of the *iThreads*, *iNumHavokThreads* and *iOpenMPLevel* settings very high doesn't automatically mean it uses that many threads - it all depends on how many threads are *actually possible* based on the information being processed.

[Cell Visibility/Loading Variables]

UGridsToLoad=5 - Increasing the *uGridsToLoad* value (always in odd steps of 5,7,9,11 etc.) will increase the number of grids around the character in which full texture data will be loaded up and visible as he/she wanders around. This clearly improves visual quality - removing blurry distant LOD textures for example when set to 11 or above - however it also noticeably increases loading times and loading pauses, introduces visual anomalies (particularly around water areas) and reduces performance significantly. Interestingly, as *uGridsToLoad* increases, the system automatically increases the *uExterior Cell Buffer* value to accommodate the greater cell data. Furthermore, if you use an even value, the engine will automatically raise it to the nearest odd value (e.g. entering 10 means the engine will set it to 11). On balance the default value of 5 is a good compromise between performance and loading pauses.

UGridDistantTreeRange=15

UGridDistantCount=25

The above settings when combined can be used to increase the visibility of distant trees. Increasing the *uGridDistantTreeRange* setting by itself has no visible impact. Yet if you also increase the *uGridDistantCount* value as well, you will start to see trees further out towards the horizon, on mountain ranges for example. Very high values will reduce performance and dramatically increase level loading times. For example experiment with a value of 200 for both to see the image quality and performance impact on your system, though I would recommend you leave these at their default for the best performance/image quality balance.

UGridDistantTreeRangeCity=4

UGridDistantCountCity=4

The above settings work in much the same way as *uGridDistantTreeRange* and *uGridDistantCount* except they apply to tree visibility within cities.

FLandTextureTilingMult=2.0000 - Decreasing this value can reducing the obvious tile pattern which occurs on land textures, but unfortunately also distorts textures close to the character. You can also use this mod for a much better fix to grass tiling.

[Tree & Grass Variables]

IMinGrassSize=120 - This setting controls the density of grass clumps. The higher the value, the less tightly packed grassy areas will be, resulting in more empty spaces in grassland, and thus noticeably higher FPS in heavily grassed areas. I suggest a value of 120 to improve FPS without overly thinning out grass.

Note: *You can also use this Low Poly Grass Mod to reduce the polygon count (complexity) of grass, boosting FPS further and making the overall look better.*

FGrassEndDistance=8000.0000

FGrassStartFadeDistance=7000.0000

The above settings control the distance at which grass ends, and the distance shortly before that which it starts to fade away. You can lower these values to increase performance, and also by reducing the difference between the FadeDistance and EndDistance.

BGrassPointLighting=0 - Setting to 1 provides more accurate lighting on grass at the cost of reduced FPS.

BDrawShaderGrass=1 - If set to 0 removes all grass which obviously improves FPS in such areas at the cost of realism.

ITreeClonesAllowed=0 - If set to a particular value, appears to determine in part the number of possible tree clones (copies of the same basic tree type) which are generated. The higher the value, the greater the potential to improve FPS and stuttering by reducing texture types and processing time, however I have not seen a major performance or visual difference from various values of this setting.

ICanopyShadowScale=512 - This value determines the appearance of tree shadow textures - lowering the resolution value (in multiples of 8) will make the tree shadows appear finer and more detailed, but this will also introduce a visible tiling pattern to the shadows. Altering this value doesn't appear to have a significant impact on FPS, but may do on older graphics cards.

BEnableTrees=1 - If set to 0, turns off all trees, boosting FPS but making things highly unrealistic outdoors.

BForceFullLOD=0 - If set to 1 uses full Level of Detail (LOD) for trees, making them appear slightly better at the cost of a small FPS drop.

[Water Variables]

BUseWaterReflectionsMisc=1

BUseWaterReflectionsStatics=1

BUseWaterReflectionsTrees=1

BUseWaterReflectionsActors=1

The above settings control the additional reflections possible on the surface of water areas. When set to 1, they allow nearby trees, objects, and other characters to reflect in the water. Note that this reduces FPS in busy areas, and that the additional reflections are only visible when you and the objects are closer to the water.

UDepthRange=125 - This option controls the depth of water visibility from above. The higher the value, the clearer the water appears, however some water glitches may also start becoming apparent, and FPS may be reduced as well.

BUseWaterDepth=1 - If set to 0, water becomes completely opaque (i.e. you cannot see into it at all from above). This can resolve some of the visual glitches, which result from changing the *uGridsToLoad* value, and can also improve FPS, at the cost of some realism.

UNumDepthGrids=3 - Also appears to control the depth of water visible, with lower values making water less transparent. If set too high this results in glitches, but if set to 1 can resolve issues with the *uGridsToLoad* value being higher than 5 and improve FPS as well.

BUseWaterLOD=1 - If set to 0 removes all water.

FSurfaceTileSize=2048.0000 - Controls the size of the water texture grids. The smaller this value, the smaller the size of texture tiles, and the faster and more tightly packed the water ripples will appear.

[Actor Variables]

BUseEyeEnvMapping=1 - If set to 0, disables environment mapping on character eyes. This makes eyes look slightly less natural and not influenced by the character's surroundings, but can improve FPS slightly.

BDisableHeadTracking=0 - If set to 1, stops characters' heads swivelling to track other characters, including your own (when in 3rd person view). Does not seem to influence performance much, and is not recommended.

BFaceGenTexturing=1 - If set to 0, Face Generator textures are removed from faces - this means they will all look smooth and lack things like age lines. There will be a slight performance improvement by doing this.

BFaceMipMaps=1 - If set to 0, character faces are slightly less detailed but FPS may improve slightly.

[Movie Variables]

SMainMenuMovieIntro=Oblivion IV logo.bik

SIntroSequence=Bethesda softworks HD720p.bik, 2k games.bik, game studios.bik, Oblivion Legal.bik

The above two settings controls the bulk of the introductory movies that are loaded up each time you start Oblivion. By setting each of these options to blank (e.g. *SMainMenuMovieIntro=*), you will prevent the intro movies from starting up, thus getting to the main menu quicker, reducing memory usage as well.

SMainMenuMovie=Map loop.bik - This is optional, but for those who also want to disable the animated map in the background of the main menu, set this option to = blank as well. This can also improve mouse responsiveness in the main menu.

[Interface Variables]

FDlgFocus=2.1000 - This setting determines the level of camera zoom on people's faces when you enter into conversation with them. A value of 4.0000 is equivalent to no zoom at all, while something like 3.0000 gives a mild zoom which I feel is better than the extreme level of zoom provided by the current setting.

BHealthBarShowing=0 - If set to 1, shows a small yellow health bar above an enemy in combat, rather than the normal curved health bar in the centre of the screen.

ISafeZoneX=5

ISafeZoneY=5

ISafeZoneXWide=5

ISafeZoneYWide=5

The above settings determine the 'safe zone' for the placement of your Heads Up Display (HUD). By reducing the values, such as to a value of 5, this will move the HUD elements at the bottom of the screen even lower, making them less intrusive.

[HDR & Bloom Variables]

BlurShaderHDRInterior

BlurShaderHDR

BlurShader

The above sections in the .ini file relate to HDR lighting effect parameters (the first two being interior and exterior HDR lighting), and the third being Bloom lighting parameters. You can alter these values in an attempt to make the lighting effects better suit your tastes. For example, some people suggest that using the following settings for the Bloom (BlurShader) lighting makes it look more 'HDR-like' without the same performance impact as real HDR:

FSkyBrightness=0.7000

fAlphaAddInterior=0.8500

FBlurRadius=0.0600

INumBlurpasses=3

IBlendType=2

Obviously you will have to experiment or find someone who has values, which you may find desirable, but note that varying these can result in strange visual glitches in certain areas, such as an oversaturated sky or odd watercolours.

Appendix A - References

Here I will list some references I used to glean information from. Most I downloaded at some point in the past just to learn more about what I was trying to do within the CS. Some are repetitive; others are unique in the information they provide. A few didn’t necessarily provide much new information for me, but may be of use to another. These are all tutorials or references put together by someone else and provided for download within the gaming community. I do not remember where I may have acquired many of them, so I am not including where to get them. Some contain this information already. There are many, many tutorials out there if one just looks for them. If one doesn’t have the info you are looking for, keeping looking, you will find it eventually.

Morrowind Mod Maker’s Manual v2.0
Compiled by Edwardsm

TESCS Wiki
Compiled and Written by many great people

Logam’s Modding Guide
Written by Logam

Povuholo's Quest Tutorial
Written by Povuholo

Oblivion Construction Set Quest Tutorial
Written by Dave Foster

There are many others to be found, but these are the one that I found to be a good start, and helped me the most.

Appendix B - Tables

In this section I will show tables from within the CS and try to provide some explanation of what they are, and possibly how to use them, or the consequences of altering data within them.

Tables will be added in future updates.

Appendix C - Websites

Here I provide a listing of websites I have found. Some are extremely useful sources of information; others are just great to visit. I list these in no particular order. Make your own judgments as to which sites are worth visiting. (If you want your website listed here, or know a good one that is not listed please contact me)

The Official Elder Scrolls Website - http://www.elderscrolls.com

The Elder Scrolls Nexus – http://www.tesnexus.com/

Planet Elder Scrolls – http://www.planetelderscrolls.com

TESCS Wiki – http://cs.edlerscrolls.com

Windfall Mod – http://windfall.50webs.com

The Lost Spires Mod – http://www.lostspires.com

The Unofficial Elder Scrolls Pages - http://www.uesp.net/index.shtml

The following sites are places you can download archive utilities. I take no responsibility for anyone using these, it is up to you if you wish to download and make use of any of these utilities:

7Zip - http://www.7-zip.org

WinRar - http://www.rarlab.com/

UltimateZip - http://www.ultimatezip.com/

WinAce - http://www.winace.com/

ZipGenius - http://www.zipgenius.it/

WinZip - http://www.winzip.com/

PKWare - http://www.pkware.com/

Appendix D - Meshes and Textures

Every object in the game that can be placed in the game world has a Mesh and Texture associated with it. When creating a new object in the CS, it must be based upon a mesh to be given physical form in the game. Textures are wrapped around the mesh to give it a ‘skin’, thus making it look like an object. Textures are associated with a specific mesh (usually assigned by the program that created the mesh, though some utilities can do this too). `.dds` files are used because they compress to smaller files well, support alpha channels, and still look pretty good. The easiest way to create a new object is to just use a mesh provided with the game. These can be found in the .bsa files in your Oblivion directory. But what if you don’t see a mesh you like? Or maybe you like the shape, but not the coloring? You want to make your own, right? No problem, there are programs you can use to do this. But I hope you have your wallet handy, you may need it.

3D Studio Max:

This software program was used to create the 3D objects and environment within Oblivion. This is expensive professional 3D-model creation software (price was around $3000, students could get it cheaper). A company called Discreet makes it. This software requires a plug-in so that it can import/export `.nif` files (3D Studio Max creates its own files, which must be converted, or exported, into the `.nif` file format for use by Oblivion).

Gmax:

GMax is another 3d modeling software program, and it’s free (get it from Discreet’s website also). Notice I said free. This software was created so modders can create objects for use in certain games. Oblivion is not one of those games. What you can do with this program is create an object, then find someone who has an appropriate version of 3D Studio Max, then beg and plead nicely asking them to convert the file into a .nif for you, then send it back to you (if someone does this, be very, very grateful). You will not be able to do any animations, nor anything advanced. But you can create simple objects that can be placed in the game world and seen. You cannot make objects that can be equipped by the PC or actors (that has to be done in 3D Studio Max). So while this is a great little free program, be aware of its limitations.

Milkshape 3D:

This is a program that can be used to create 3D models by Chumbalum Soft. You can use it to create meshes for Oblivion; but keep in mind; it wasn’t made for Oblivion. Results may vary based upon what you are creating with it. There is a .nif exporter, get it from NetImmerse File: Liberation Association (NIFLA). The exporter is still in development,

---

but at this time it cannot physique some objects, do transparency, animations, or particle effects. Again, I do not have nor use this software, so cannot verify exactly how useful it is, or any limitations inherent in using it.

PhotoShop:

PhotoShop is a program made by Adobe. It can be used to retexture, or skin an image file (or create a new one). It seems to be quite popular, and quite expensive. To create `.dds` files you must get a plug-in with which you will be able to export your file to the `.dds` format. You want to find the plugin?

PaintShopPro:

PaintShopPro is a program originally made by Jasc, but has been acquired by Corel. It too can be used to retexture an image. PaintShopPro by all accounts works just as well as PhotoShop, but is far cheaper. Reportedly, the `.dds` exporter for PhotoShop also works fine with PaintShopPro. However, that was with prior versions of the program, I cannot verify if it still works with current versions. You want to find the plugin?

The Gimp:

The Gimp is an image-editing program that can also be used to retexture image files. It is a free program. If you are not conversant with using this program, I suggest you look at this online manual called Grokking The Gimp on how to use it.

Blender 3D:

This is a program that can be used to create 3D models for Oblivion; you do need an exporter/importer plugin though.

Appendix E - Contacting

I would greatly appreciate any comments on this work, even negative. Anything I can use to improve it. If you wish to do so, email me. I will make every effort to reply as soon as possible.

Email comments, feedback, mistakes or anything you wish about the manual to:

Lord_Gannondorf@hotmail.com

Good luck and happy modding!