

# Scripting Tutorial: The Basics

---

## **Table of Contents:**

1. [Scripting Tutorial: The Basics](#)
2. [Getting Started](#)
3. [Before you start](#)
4. [OnPostInit](#)
5. [OnPreCacheObjects](#)
6. [Objective 1](#)
7. [Objective 2](#)
8. [Objective 3](#)
9. [Cutscenes: 101](#)
10. [Final Word](#)
11. [Tips](#)

## Scripting Tutorial: The Basics

So, you want to make your very own mod? Then you've come to the right place. In this brief tutorial, I will attempt to teach you the very basic skills you will need to at least begin your very own dream mod or single mission story. To begin scripting, you will need the following tools:

- 1 The FF Scripting document
- 2 A python Editor (notepad, pythonWin, Crimson editor)
- 3 A basic story or outline
- 4 FFEedit for map editing and character editing (not covered)

## Getting Started

First, you will need a basic story from which you will work from. You can change this story over time, but at least have an idea in your mind about what you want to accomplish. This makes scripting a little easier and more linear, so the complexity level is severely dropped. I commonly make a semi-detailed outline and describe each objective I want the player to accomplish in notepad before I do anything else.

EXAMPLE:

Story: Batman believes that a warehouse holds the secrets to one of Joker's most deadly plans to enslave the citizens of Gotham!

- Obj. 1- Explore the hideout
- Obj. 2- Collect the secret plans
- Obj. 3- Escape before the building explodes

In my experience, making a brief outline tends to keep me on track and helps me pick out which commands I will be using. So let's move on to the next section.

## Before you start

Before you actually jump into scripting out your story, you will need to import the following files at the top of your document:

```
from cshelper import *
from js import *
import event
```

These commands basically allow you to use the other commands without using the prefixes of “js” or “cs”. I won’t be covering how to get FFX to work in your mod/mission, but it’s just as easy. Under this section, you may even add your mission description and objective notes. Make sure that you use the ‘#’ symbol before each comment. Now let’s move on.

## OnPostInit

You’ve made it this far, so stay with me. This is one of the most important sections in the scripting process. This section is basically the starting point of the mod. It is also used to set up some things that will happen later in the mission. Some examples of the things that may go in this section include: regDeath, regTimer, Mission\_SetAttr, play, any camera commands, and sunlight control; just to name a few.

I’m working from my outline, which I provided earlier in this tutorial, so yours might not resemble mine. I’ll try to keep mine simple. In this section, I want the screen to appear black (Camera\_Fade) and I want a cut scene to play to set up the basic story before I allow the player to do anything. I will also set up a command to keep track of 5 thugs that will challenge batman and lead into objective 2. Most importantly, I want the camera to be focused on Batman. The simplest command to use to achieve this would be to use the “look” command.

EXAMPLE:

```
#Example script

from cshelper import *
from js import *
import event
```

```
#####  
#Obj. 1- Explore the hideout  
#Obj. 2- Collect the secret plans  
#Obj. 3- Escape before the building explodes  
#####  
  
def OnPostInit():  
    print 'Batman begins...'  
    Camera_Fade(0)  
    look('batman')  
    play(start)  
    regDeaths('thug_with_bat', 'thugs_dead')  
    Mission_SetAttr('thugs_count', 0)
```

Hopefully, you have the scripting document open as you do this. I will explain the commands as best I can, but having the document open and looking at how the commands are set up will help you as well. The print command does exactly what you might think. It prints the message you inputted in the single quotations to the script.log file in your FFVTTR folder. This command is very helpful in debugging as it helps you locate problem areas or to pick out which defined events aren't launching.

Camera\_Fade is another command that does exactly what you might think it does. The interval in the parentheses is the amount of time it takes for the camera to fade to black. In this example, it is set to 0 so that when the mission loads, it will already be dark.

The look command focuses the camera on batman or any object you specified within the single quotes in the parentheses. It has four more argument fields, but they do not have to be used if you are not comfortable with the camera controls yet.

The play command starts a cutscene. In this case, it will start a cutscene named "start".

regDeaths and Mission\_SetAttr are two very important commands when dealing with the number of KOs to a certain group of objects, in this case, thugs with bats. regDeath will register the deaths of the thugs and will call on the yet to be defined event 'thugs\_dead' when one dies. Mission\_SetAttr sets the attribute 'thugs\_count' to 0. This will be very important later on when we want Objective 1 to be completed after all 5 thugs are KO'd.

Ok, I think we're done with the section, so; I think we should move on to the next section.

## OnPrecacheObjects

This section basically precaches objects (usually characters) on the map before the game starts. Only one command can be used here and it is “Mission\_PrecacheTemplateObj()”. All this command does is precaches data associated with an object so that it can be spawned without the game pausing. For my example, I will only need to use this command twice: Once for Batman and once for the five thugs (because all the thugs use the same template).

Example:

```
Mission_PrecacheTemplateObj('batman')
Mission_PrecacheTemplateObj('thug_with_bat')
```

It should be noted that the names used in the quotes are the template names obtained from FFedit. Alrighty then, we're done with this section. My script now looks like this:

```
#Example script

from cshelper import *
from js import *
import event

#####
#Obj. 1- Explore the hideout
#Obj. 2- Collect the secret plans
#Obj. 3- Escape before the building explodes
#####

def OnPostInit():
    print 'poor MJ'
    Camera_Fade(0)
    look('batman')
    play(start)
    regDeaths('thug_with_bat', 'thugs_dead')
    Mission_SetAttr('thugs_count', 0)

def OnPrecacheObjects():
    print 'precache'
    Mission_PrecacheTemplateObj('batman')
    Mission_PrecacheTemplateObj('thug_with_bat')
```

I added another print command just for debugging purposes. From here on out, there's no required section. What goes next is totally dependent on the scripter. I suggest working on one objective at a time and going from there, but do whatever feels comfortable to you.

## Objective 1: Defining an Event

Looking back over my notes, my first objective was for the player to explore Joker's hideout. I later added in the thugs and required the player to beat all 5 of them before moving on to objective 2. So now we're getting into our very first defined event! Remember that little paragraph about `Mission_SetAttr` and `regDeath`? Here's where they will do their work.

```
regDeaths('thug_with_bat', 'thugs_dead')
Mission_SetAttr('thugs_count', 0)
```

I named my event 'thugs\_dead' so that's what my next event will be. I want this event to count the number of thugs KO'd and when that number reaches 5, I want the objective to be completed and a new objective to appear to the player as well as spawning the object to complete the second objective. So, let's set that up:

```
def thugs_dead(event):
    print 'thug count'
    tKO = Mission_GetAttr('thugs_count') + 1
    Mission_SetAttr('thugs_count', tKO)
    if tKO == 5:
        print 'all thugs defeated'
        Objective_Complete('primary1')
        Object_Spawn('plans', 'plans', 'plans_positional')
        Objective_Add('primary2', 'Collect the secret plans', 0, 10)
```

Wait! Don't freak out, ok? You'll find that this block of code is pretty easy to understand. So let the fun begin. First, 'def' means define. We want to define the event we set up way in `OnInit`. I once again used the `print` command for debug purposes to make sure the event is being set off.

The 'tKO' part is basically a variable you define. It can be anything you want. I stick with the 3 letter variable, but that's just because I'm comfortable with it. `Mission_GetAttr()` is used to get information from a previously established attribute ('thugs\_count') and to add 1 to it after a thug dies.

The `Mission_SetAttr` command used here is to help synchronize the variable with the attribute. Think of it as a simple algebra problem. Ok, don't freak out, it's simple math. The 'tKO' variable is equivalent to X while the attribute ('thugs\_count') is equivalent to 0 and you're trying to reach the total of the number of thugs on the map (5). The equation would look like this:  $X + 0 = 5$ .

When my 'equation' equals out to 5 (python uses two "=" to represent equal to), the following events will happen next:

The game will print 'all thugs defeated' to my log files, complete the first objective (which will be defined in the first cutscene later in this tutorial), spawn an object named

'plans' using the 'plans' template at the positional marker on the map named 'plans\_positional' and add a new objective. I'll leave it up to you to figure out how to set up "Objective\_Add" since it's not hard and clearly explained in the scripting document. And that's it for the first event. Now, that wasn't hard was it? This is what I have now:

```
#Example script

from cshelper import *
from js import *
import event

#####
#Obj. 1- Explore the hideout
#Obj. 2- Collect the secret plans
#Obj. 3- Escape before the building explodes
#####

def OnPostInit():
    print 'poor MJ'
    Camera_Fade(0)
    look('batman')
    play(start)
    regDeaths('thug_with_bat', 'thugs_dead')
    Mission_SetAttr('thugs_count', 0)

def OnPrecacheObjects():
    print 'precache'
    Mission_PrecacheTemplateObj('batman')
    Mission_PrecacheTemplateObj('thug_with_bat')

def thugs_dead(event):
    print 'thug count'
    tKO = Mission_GetAttr('thugs_count') + 1
    Mission_SetAttr('thugs_count', tKO)
    if tKO == 5:
        print 'all thugs defeated'
        Objective_Complete('primary1')
        Object_Spawn('plans', 'plans', 'plans_positional')
        Objective_Add('primary2', 'Collect the secret plans', 0, 10)
```

## Objective 2: Collecting items

For objective 2 of the mission, Batman will need to collect the secret plans we spawned earlier. Well to do this, we will need to make some minor edits to the event we just defined. Basically, all we need to do is add the `Mission_CustomAction()` to the end of the defined event.

```
def thugs_dead(event):
    print 'thug count'
    tKO = Mission_GetAttr('thugs_count') + 1
    Mission_SetAttr('thugs_count', tKO)
    if tKO == 5:
        print 'all thugs defeated'
        Objective_Complete('primary1')
        Object_Spawn('plans', 'plans', 'plans_positional')
        Objective_Add('primary2', 'Collect the secret plans', 0, 10)
```

Will now become:

```
def thugs_dead(event):
    print 'thug count'
    tKO = Mission_GetAttr('thugs_count') + 1
    Mission_SetAttr('thugs_count', tKO)
    if tKO == 5:
        print 'all thugs defeated'
        Objective_Complete('primary1')
        Object_Spawn('plans', 'plans', 'plans_positional')
        Objective_Add('primary2', 'Collect the secret plans', 0, 10)
        Mission_CustomAction('collect plans', 'batman', 'secret_plans',
'fnc_plans', 5, 0)
```

I'm relying on you to read up on the `Mission_CustomAction` command in the scripting docs to learn how to set it up. Next, we will use your newly acquired event defining skills to define yet another, simpler event. My even will be named `fnc_plans`. After the plans are collected, I want the event to destroy the plans to simulate Batman picking it up, complete the objective, add the final objective, and set up another event to cause batman trouble, in this case a countdown to an explosion.

```
def fnc_plans(plan, hero):
    print 'play cutscene, destroy item'
    destroy('secret_plans')
    Objective_Complete('primary1')
    regTimer('explosion', 300)
    Objective_Add('primary3', 'Escape the building', 0, 10)
    regHeroMarkerEnter('escape', 'Mission_Win')
```

You may have noticed that instead of `def fnc_plans(event)` I used `(plan, hero)`. That's because those are the two objects that will be interacting and I only want the event to be set off when the player clicks on the plans. Again, I used the `print` command for debugging. I also used the `destroy` command, but that's self explanatory, I believe. The only new commands here are `regTimer` and `regHeroMarkerEnter`. Again, I will leave it

up to you to look at how they're set up. Here, I have regtimer set to activate the yet to be defined event, 'explosion' when 300 seconds, or 5 minutes, are up. regHeroMarkerEnter is used to register the entering of the marker named 'escape' by a hero which will set off another event named 'Mission\_Win'. For this to work correctly, you have to make sure the marker has a large enough area for the hero to run through to activate the event.

## Objective 3: Defining more events!

Ok, so we're almost done with our mission, minus the simple cutscene we will do. First, let's set up the explosion event:

```
def explosion(event):
    print 'something explodes'
    explode('bomb', 'generic fire explosion', intensity = 3)
    Mission_Lose()
```

Short and sweet, huh? On my map, I will have a hidden object named bomb, but you can spawn in the bomb after the completion of objective 2 if you please. The explode command is pretty easy to use and once again, you will need to use the scripting docs to see how it's set up. Mission\_Lose() is also self explanatory. Now the other event:

```
def Mission_Win(event):
    print 'game over'
    Mission_Win()
```

And we're done with the core part. My final script looks like this:

```
#Example script

from cshelper import *
from js import *
import event

#####
#Obj. 1- Explore the hideout
#Obj. 2- Collect the secret plans
#Obj. 3- Escape before the building explodes
#####

def OnPostInit():
    print 'poor MJ'
    Camera_Fade(0)
    look('batman')
    play(start)
    regDeaths('thug_with_bat', 'thugs_dead')
    Mission_SetAttr('thugs_count', 0)
```

```

def OnPrecacheObjects():
    print 'precache'
    Mission_PrecacheTemplateObj('batman')
    Mission_PrecacheTemplateObj('thug_with_bat')

def thugs_dead(event):
    print 'thug count'
    tKO = Mission_GetAttr('thugs_count') + 1
    Mission_SetAttr('thugs_count', tKO)
    if tKO == 5:
        print 'all thugs defeated'
        Objective_Complete('primary1')
        Object_Spawn('plans', 'plans', 'plans_positional')
        Objective_Add('primary2', 'Collect the secret plans', 0, 10)
        Mission_CustomAction('collect plans', '', 'secret_plans',
'fnc_plans', 5, 0)

def fnc_plans(plan, hero):
    print 'play cutscene, destroy item'
    destroy('secret_plans')
    Objective_Complete('primary1')
    regTimer('explosion', 300)
    Objective_Add('primary3', 'Escape the building', 0, 10)
    regHeroMarkerEnter('escape', 'Mission_Win')

def explosion(event):
    print 'something explodes'
    explode('bomb', 'generic fire explosion', intensity = 3)
    Mission_Lose()

def Mission_Win(event):
    print 'game over'
    Mission_Win()

```

Only thing left to do is to set up a very simple cutscene. To do this, you will need to navigate to your mod's Lang/English folder and open the captions text file. You will need to add in your own dialogue. I will use the following:

MISS\_1\_B1, neutral, A warehouse in the middle of nowhere.  
MISS\_1\_B2, neutral, This must be Joker's current hideout  
MISS\_1\_B3, neutral, Time to explore it and find out what dastardly scheme he's plotting  
MISS\_1\_B4, neutral, Something important goes here.

Oh, the format for adding lines to captions is usually: MISSPCH\_(Mission Number)\_(Character speaking)\_(Line number)

But I use my own variation of this format. Once you are done adding lines, save the document and open FFEEdit. Make sure the primary data path is pointing to your mod folder and the Output directory is pointing to your mod's language\english folder and press the button labeled "Generate Language Files". While you wait for that, you can begin thinking out your cutscene.

## Cutscenes: 101

The final section of any mod is the cut scenes. This part is considered to be one of the easier parts. Cut scenes can do a variety of things and can set up things for later objectives. A cutscene can be very complex and have fx and animations in them, or just simple dialogue. We'll go simple for right now. Oh yeah, remember objective 1? We never added it. We'll add it here:

```
start = [
(
    "startCS()",
    "see()",
    "peace()",
    "wait(1)",
    "speak('batman', 'MISS_1_B1')",
),
(
    "Camera_UnFade(1)",
    "wait(1)",
),
(
    "speak('batman', 'MISS_1_B2')",
),
(
    "speak('batman', 'MISS_1_B3')",
),
(
    "speak('batman', 'MISS_1_B4')",
),
(
    "endCS()",
    "noLook()",
    "AI_CancelAllSpeech()",
    "Objective_Add('primary1', 'explore the hideout', 0, 10)",
    "Mission_SelectHero('batman')",
    "war()",
),
]
```

Whew, cut scenes are a lot of work, but extremely easy to put together. I'll try to go block by block. To start a cutscene, you first need to name it exactly like you named it in the play() command. I named mine start. So that's how I got: Start = [

After every block, you will use a combination of ),(. If you're missing any of these, it'll result in an error. Also, every command is now enclosed in double quotations and requires a comma after the last quote. Example: "peace()",

Now that that's out of the way, let's tell what each command in the first block does.

```

start = [
(
    "startCS()",
    "see()",
    "peace()",
    "wait(1)",
    "speak('batman', 'MISS_1_B1')",
),
(
    "Camera_UnFade(1)",
    "wait(1)",
)
]

```

startCS() is a self-explanatory command. It starts the cutscene. The see() command allows villains to be seen if they are participating in the cutscene. Peace() disables enemy AI. Wait() will tell the game to pause for 1 second before carrying out the next event. This command will be used a lot in cut scenes after most of the other commands. Speak('character', 'dialogue tag') will make the selected character speak the dialogue that corresponds with the tag of the dialogue. In this case, 'MISS\_1\_B1', will make the character, Batman, speak the line "A warehouse in the middle of nowhere."

The next unknown command is Camera\_UnFade(#). It makes the screen unfade from black in the number of seconds specified. Finally, to end a cutscene, you will need to use endCS(). The noLook() command returns camera control back to the player. AI\_CancelAllSpeech() cancels any speech that may be playing. Mission\_SelectHero() basically selects a hero for the player. You don't have to use this command. Finally, war() reactivates the enemy AI.

## Final word

By now, you should have a working simple mission to play with. It's not big or complex, but it's playable. Now that you know the basics, maybe you will try your hand at making your mission more complex and difficult. You might even attempt to add more complex, visually appealing cut scenes to your mod. Heck, you might discover an even easier way to do what I just showed here. Whatever you do, have fun with it. Here's my final script:

```

#Example script

from cshelper import *
from js import *
import event

```

```
#####  
#Obj. 1- Explore the hideout  
#Obj. 2- Collect the secret plans  
#Obj. 3- Escape before the building explodes  
#####  
  
def OnPostInit():  
    print 'poor MJ'  
    Camera_Fade(0)  
    look('batman')  
    play(start)  
    regDeaths('thug_with_bat', 'thugs_dead')  
    Mission_SetAttr('thugs_count', 0)  
  
def OnPrecacheObjects():  
    print 'precache'  
    Mission_PrecacheTemplateObj('batman')  
    Mission_PrecacheTemplateObj('thug_with_bat')  
  
def thugs_dead(event):  
    print 'thug count'  
    tKO = Mission_GetAttr('thugs_count') + 1  
    Mission_SetAttr('thugs_count', tKO)  
    if tKO == 5:  
        print 'all thugs defeated'  
        Objective_Complete('primary1')  
        Object_Spawn('plans', 'plans', 'plans_positional')  
        Objective_Add('primary2', 'Collect the secret plans', 0,  
10)  
        Mission_CustomAction('collect plans', '', 'secret_plans',  
'fnc_plans', 5, 0)  
  
def fnc_plans(plan, hero):  
    print 'play cutscene, destroy item'  
    destroy('secret_plans')  
    Objective_Complete('primary1')  
    regTimer('explosion', 300)  
    Objective_Add('primary3', 'Escape the building', 0, 10)  
    regHeroMarkerEnter('escape', 'Mission_Win')  
  
def explosion(event):  
    print 'something explodes'  
    explode('bomb', 'generic fire explosion', intensity = 3)  
    Mission_Lose()  
  
def Mission_Win(event):  
    print 'game over'  
    Mission_Win()  
  
####cutsscenes####  
  
start = [  

```

```
(
    "startCS()",
    "see()",
    "peace()",
    "wait(1)",
    "speak('batman', 'MISS_1_B1')",
),
(
    "Camera_UnFade(1)",
    "wait(1)",
),
(
    "speak('batman', 'MISS_1_B2')",
),
(
    "speak('batman', 'MISS_1_B3')",
),
(
    "speak('batman', 'MISS_1_B4')",
),
(
    "endCS()",
    "noLook()",
    "AI_CancelAllSpeech()",
    "Objective_Add('primary1', 'explore the hideout', 0, 10)",
    "Mission_SelectHero('batman')",
    "war()",
),
]
```

## Tips

1. If your python editor has it, check your documents for any errors with the “check module” command.
2. If you’re having a problem, you should try asking the FR boards for help or stop scripting for about an hour and try to fix the problem again.
3. If you want to advance your scripting knowledge and skills, start small and work your way up.
4. always check your spacing.

## Further Reading

If you're having trouble with FFEDIT or editing a map, I suggest you take a look at this older tutorial for ff1. Everything is pretty much the same in FFVTTR as well: [FFEdit Tutorial](#)

Also, I have also included a link to my python file if you need an example with correct formatting. It can be found here: [Example python document](#)