

Tutorial Master – Instruction Manual

Version v1.01 (August 2016)

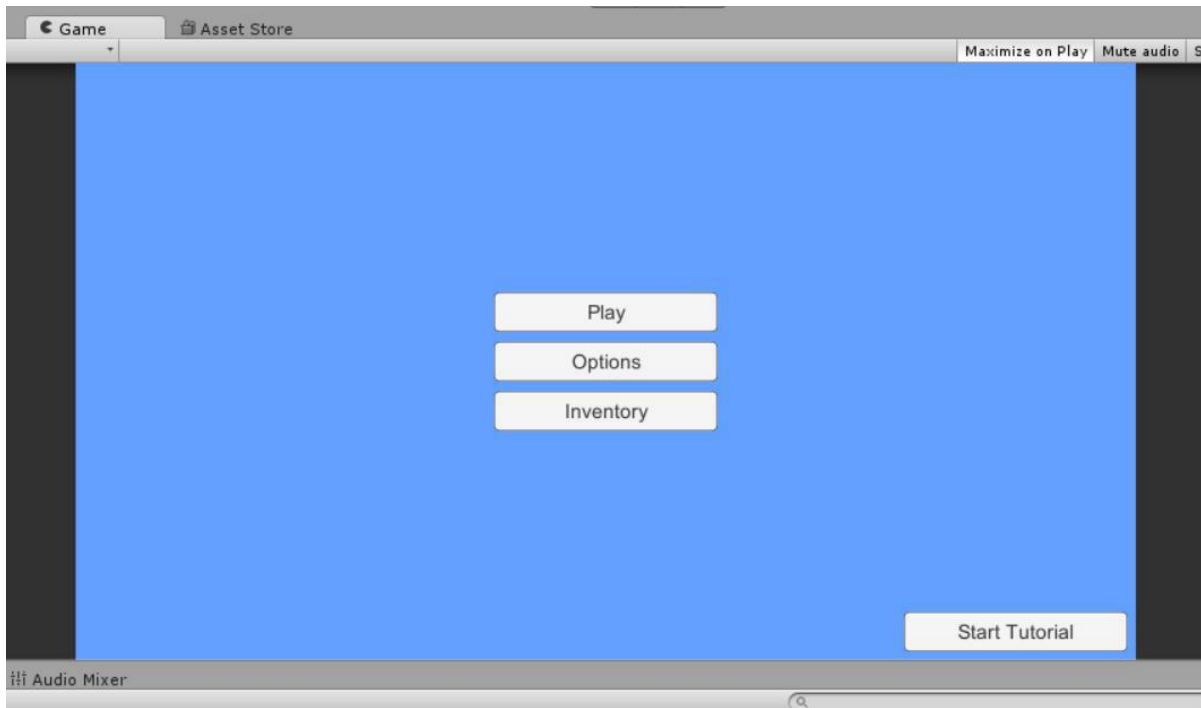
Getting Started

Assuming you have Tutorial Master imported into your project, let's get started!

Step 1

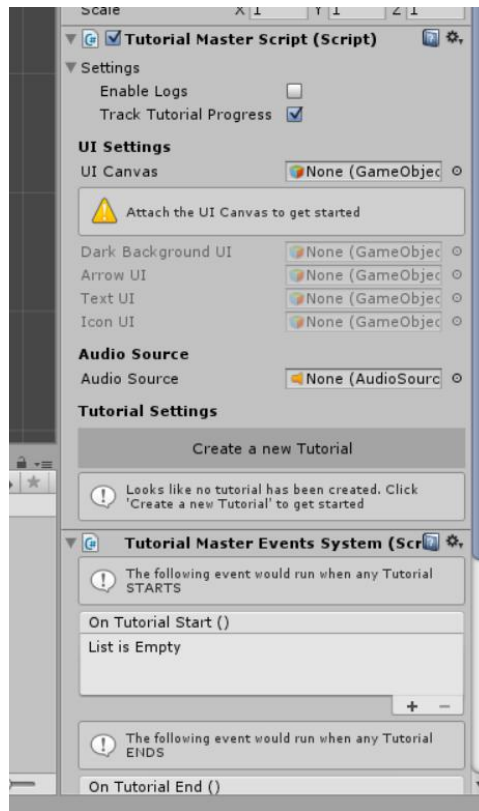
1. Create Canvas and add UI buttons to it (3 or more) – you can skip this step if you already have UI to work with
2. Add button “Start Tutorial”. We'll use it to trigger Tutorial to start

At this point, this is how our Unity scene should look like:



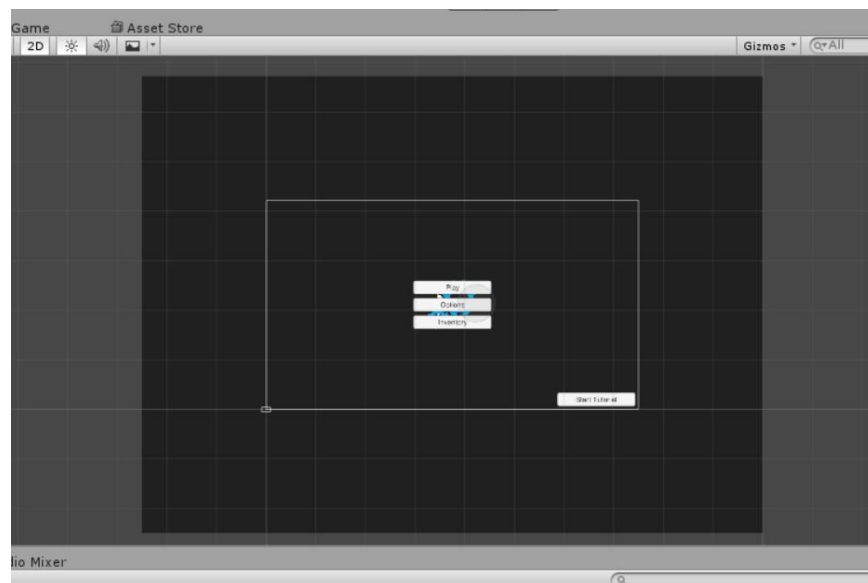
Step 2

1. Create an empty game object and name it whatever you like (e.g. “Tutorial Master Manager”). Alternatively, go to **Window > Tutorial Master > Generate a Tutorial Master GameObject**
2. Then, add a “Tutorial Master” Component to it the empty GameObject
3. You’ll see a custom inspector – the heart of Tutorial Master



Step 3

1. Attach the Canvas to the **UI Canvas** field
2. You can now add necessary UI components. You may have also noticed that a “Generate Missing Components” button has appeared. Press it and you’ll see that missing components have filled up.

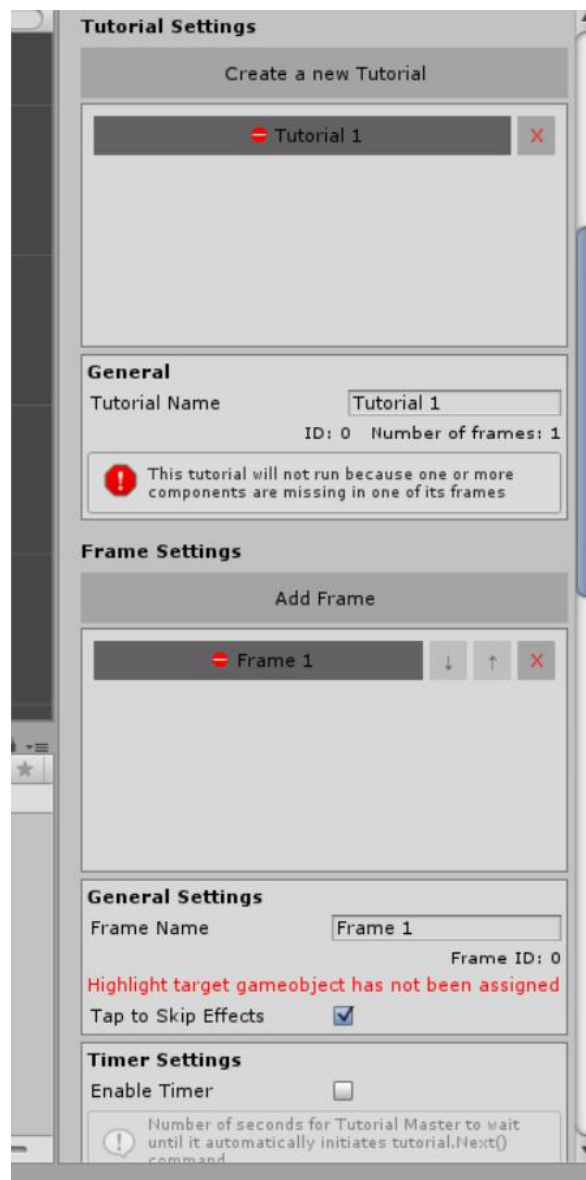


3. Reposition the Dark Background UI to fit the whole Canvas. We must make sure it fills the whole screen. When done, disable it
4. Move the Arrow UI away and disable it - nothing needs to be done to it. Feel free to change the sprite to your own or change colours (**make sure that the arrow faces right**)
5. Position Icon UI and Text UI wherever you want to. It will be used to show instructions to our player. Disable them as well.

Now, with our UI configured, it's time to tweak Tutorial Master!

Step 4

1. Press "Create a New Tutorial". Give it a name if you want (e.g. Newbie Tutorial). Then press "Add Frame". You should now see a lot of options:



This is where you're going to spend most of your time with Tutorial Master. All options are available: effects, audio etc.

Feel free to play around with it. Add a few frames with different settings. Here's the list of settings and their meaning if you don't understand something.

General Settings



General Settings

Frame Name Frame ID: 0

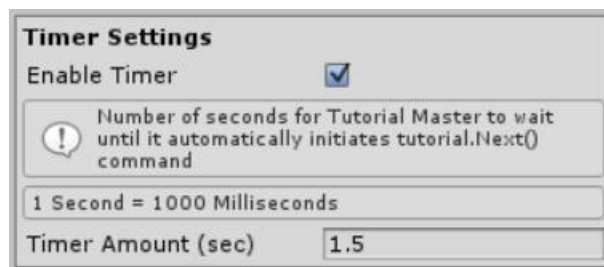
Tap to Skip Effects ☒

Frame Name – name the frame has. Makes it easier to identify other frames from each other. You may leave it as is, it won't affect the tutorial.

Frame id – shows the id of the current frame.


Tap to Skip Effects – if set true, while in game, if you click anywhere, the current animation (icon fading in, text flying in etc.), will be skipped.

Timer Settings



Timer Settings

Enable Timer ☒

 Number of seconds for Tutorial Master to wait until it automatically initiates tutorial.Next() command

1 Second = 1000 Milliseconds

Timer Amount (sec)

Enable Timer – the timer will automatically go to the next frame after X seconds has passed

Timer Amount (sec) – how many seconds to wait till to go to next frame

Dark Background Settings

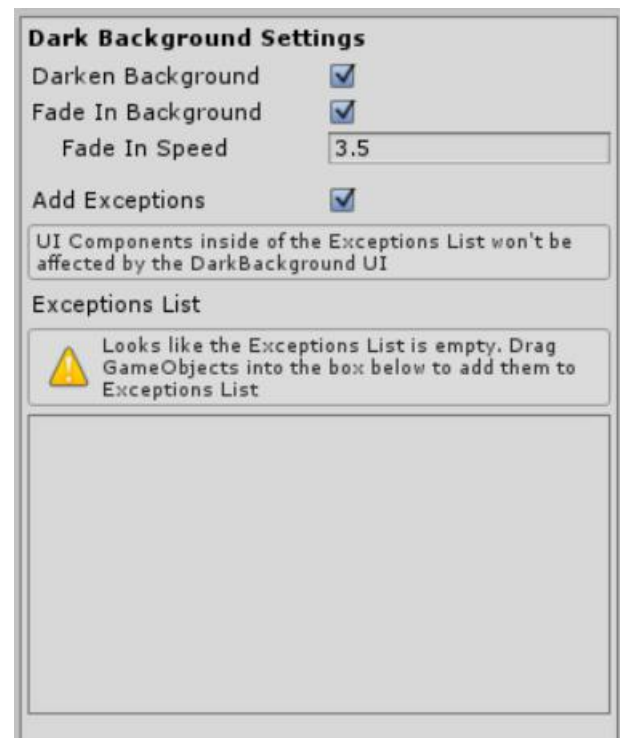
Darken Background – if set true, dark background will appear, covering everything except arrow, icon, text, highlight target and exceptional objects (if specified)

Fade In Background – the dark background will fade in when the frame is being played. Recommended to be used once at the beginning of the tutorial for full effect

Fade Speed – how fast do you want background to fade-in?

Add Exceptions - if set true, dark background will ignore some of the selected objects

Exceptions List – GameObjects that would be ignored by the dark background. Add them by dragging them into the window



Dark Background Settings

Darken Background ☒


Fade In Background ☒

Fade In Speed

Add Exceptions ☒

UI Components inside of the Exceptions List won't be affected by the DarkBackground UI

Exceptions List

 Looks like the Exceptions List is empty. Drag GameObjects into the box below to add them to Exceptions List

Highlight Settings

Highlight Target – do you want to highlight target in this frame?

Target – UI GameObject that you wish to be highlighted

Detect UI Clicks – this option is available only if your UI target is a button (or has a Button component attached to it). If set true, if player presses the button, it will go to the next frame. It does not affect current OnClick() events the button has.

Point Arrow – if set true, an arrow will be pointing at your UI target

Pointing Direction – from which direction you want the button to be pointing?

Position Offset – if you want to customize the distance between the UI target and arrow

Floating Effect – if set to true, the arrow will have floating effect

Floating Range – the bigger the range, the further maximum distance arrow will float from UI target

Floating Speed – how fast arrow floats

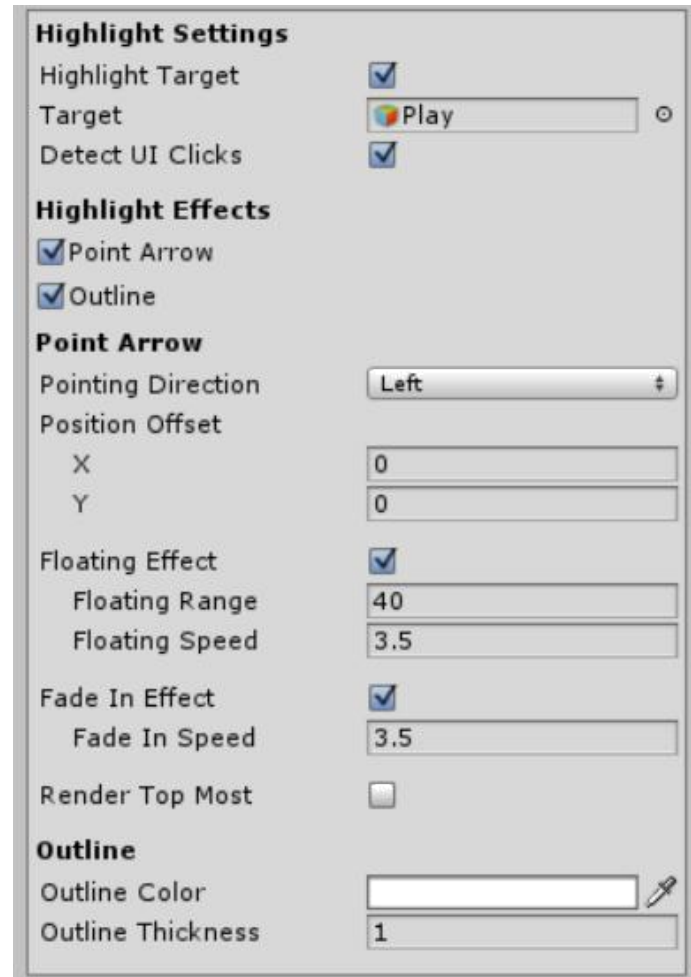
Fade In Effect – if true, arrow will have a fade-in effect when the frame starts

Render Top-Most – if true, arrow will be rendered on top of everything

Outline – if set to true, an outline will be drawn around the UI

Outline Colour – determines what colour the outline is going to be

Outline Thickness – how thick the outline is



The image shows a 'Highlight Settings' panel with various options and values:

- Highlight Target**: ☒
- Target**: Play (with a play button icon)
- Detect UI Clicks**: ☒
- Highlight Effects**:
 - ☒ Point Arrow
 - ☒ Outline
- Point Arrow**:
 - Pointing Direction**: Left (dropdown menu)
 - Position Offset**:
 - X: 0
 - Y: 0
- Floating Effect**: ☒
 - Floating Range**: 40
 - Floating Speed**: 3.5
- Fade In Effect**: ☒
 - Fade In Speed**: 3.5
- Render Top Most**: ☐
- Outline**:
 - Outline Color**: (color picker icon)
 - Outline Thickness**: 1

Text and Icon

Use Icon – if set true, an icon will be displayed where Icon UI is residing

Icon Sprite – select what icon you want it to be

Fly In – if set true, the icon/text will fly into the scene

Fly Speed – how fast do you want icon/text to fly in?

Warp Value – how far away to teleport icon/text from its current position (to be calculated automatically in future release)


Fly In Direction – from which direction should icon/text fly into the scene?

Use Text – if set true, a text will be displayed where Text UI is residing

Description Text – these should contain instructions to the player.
E.g. “Press this button to continue”

Text and Icon Settings

Use Icon ☒

Icon Sprite 

Appearance Effects

☒ Fly In ☒ Fade In

Fly In

Fly Speed

Icon Warp Value

Fly In Direction

Fade In

Fade In Speed

Use Text ☒

Description Text

Amazing Text!

Appearance Effects

☒ Fly In ☒ Fade In

Fly In

Fly Speed

Text Warp Value

Fly In Direction

Fade In

Fade In Speed

Audio Settings



Play Audio – if set true, audio clip will be played when the frame starts

Audio Clip – what audio clip do you want to be played?

Expose Settings – if true, additional audio settings will be used

Mixer Group Output – which mixer group do you want audio source to belong to?

Loop – if true, audio clip will loop

Volume – change the volume of the audio clip. 1 is default. 0 is mute

Once you're done creating your frames and tweaking settings, it's time to make it run!

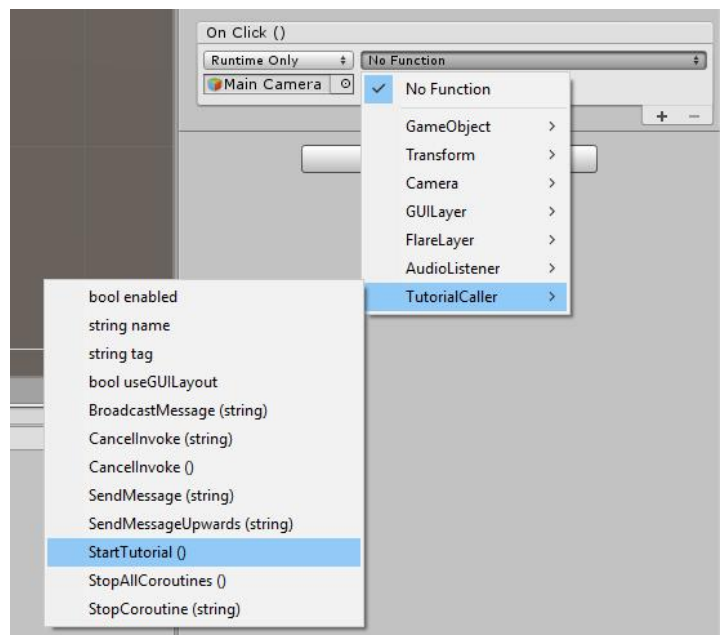
Step 5

1. Create a new script and call it whatever you want it to be (e.g. "TutorialCaller"). And attach it to some GameObject (e.g. Main Camera)
2. Open it, and add `using HardCodeLab.TutorialMaster;` Now, we'll be able to access all Tutorial Master functionalities
3. Create a new public void `StartTutorial()`
4. Inside of it, call `tutorial.Start(0)`. Because we have only one tutorial, it has an ID of 0.
5. Optionally, you may remove `Update()` and `Start()` voids
6. Save it and let it compile.

Here's my code for reference:

```
using UnityEngine;
using System.Collections;
using HardCodeLab.TutorialMaster;

public class TutorialCaller : MonoBehaviour
{
    public void StartTutorial()
    {
        tutorial.Start(0);
    }
}
```



Step 6

1. Select your “Start Tutorial” button and add an On Click () event.
2. Attach GameObject where your script resides and select our function

With that done, you’re ready for a test run!

Play the game, press “Start Tutorial” and everything should now work.