First Person Story Adventure Template for Unreal Engine 4

Documentation

Dennis Pauly, 2019

www.dennispauly.com
contact@dennispauly.com

Content

Preface

Thank you for checking out my template, I hope it's of good use for you.

This documentation should include everything you need to know to use the template. If you have any questions, found a bug or ran to any other issue with it, please feel free contact me. And please, before leaving a bad rating, tell me how I can improve. Please also check the Discord server for help: https://discord.gg/KeBSxhK

If you're happy with the template, **please help me by leaving a good rating** on the marketplace page.

Also, I'm always interested in your suggestions and wishes for future updates as I'm continuously working on improving this package. Leave a comment under the product info or write a mail to: contact@dennispauly.com

Thank you for buying / your interest in this template!

Cheers,
Dennis (palinoia gamedev)

PS: If you like what I'm doing and want to support me, please check out my Ko-fi page, as asset development is so much easier with coffee:

1. Personalization

Inside the folder Blueprints->Settings you'll find two structures: BP_StoryAdv_Settings and BP_UserInterfaceIcons. In the first structure, you can adjust general settings like trace distance, depth of field, etc. BP_UserInterfaceIcons stores button icons for gamepad and keyboard keys. You can change the icons inside the structure, so you don't have to change it inside the widget.

Note: Please only change the default values, not the structure itself (top area inside the structure).

2. Setting up your Character

Simply replace the UE Mannequin with your own character mesh. You can also clear the mesh, if you don't want to have a character mesh. In this case you may need to unparent the camera first. The camera is parented to the head bone of the character mesh, so it follows the movement of its head animation.

The included character blueprint includes a simple sprint event. Copy it to your own character, if you like.

3. Configure Input

All needed input is already set up for both mouse and keyboard and gamepad. Do some changes if you don't like the default key bindings. Keep in mind that you will have to change some control hints within the user interface.

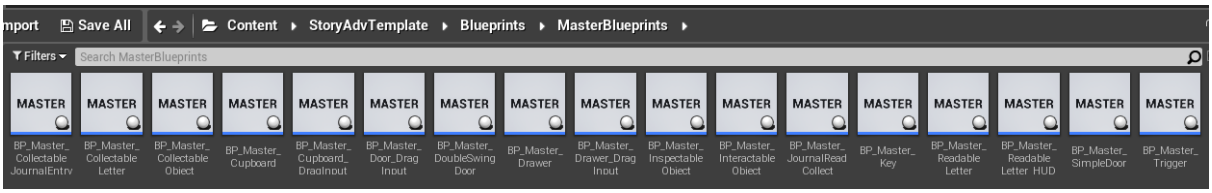
4. Game Controller Blueprint

This blueprint stores all important data such as inventory and HUD information. Be sure you have this actor placed in your persistent level and that there's only one of these actors. **Also, don't change this blueprint, so you can replace it after template updates.** If you want to make additions (for this and all other blueprints provided by the template) **create child blueprints first and add your code there. This way you can still update all master blueprints.**

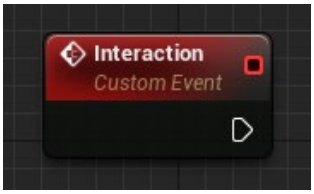
5. Interaction Outlines

To enable outlines for interactive objects, you need to have a Post Process Volume in your level. Add a Post Process Material to this volume and assign PP_ObjectOutline to it. You can change parameters like outline color and thickness by opening this material.

6. Master Blueprints



This template contains various master blueprints you can create child blueprints of (or use the example child blueprints). One of them, BP_Master_InteractableObject is the basic blueprint used for interaction. All other blueprints are children of this "main" master blueprint. **If you want to create a new interactive object**, for example a lamp, you may create a child blueprint out of BP_Master_InteractableObject. If you want to create a new door blueprint, create a child from BP_Master_SimpleDoor, BP_Master_DoubleSwingDoor or BP_Master_Door_DragInput as it contains all you need for the specific kind of door. All you need to do is set up your mesh and place it into your level. You can of course also duplicate an example blueprint.



All these blueprints (except BP_Master_Trigger) contain a component called Object and an event called Interaction. This event will be called when clicking on the containing Object Component. An interactable actor can also have multiple Object components. Duplicate the inherited object or create a new one with the tag "Interactable". The object must have collision enabled and block line traces at least. If you want to create a new trigger, you may create it as a child of BP_Master_Trigger, as all children of this blueprint are saved in the save game file.

Note: The default Object doesn't have to be interactable. If you delete it's tag it's just a static mesh you can use as frame or container.

7. Inspectable Objects

The X-Y-Pivot an inspectable object is rotated in hand is the X-Y-Pivot of the actor (every object will be placed in the middle of its Z-Axis automatically). Place your object relative to the scene to change the inspection pivot as in the example actor blueprint. You can also change the distance an object is inspected individually in the details panel (Inspect Distance). This number represents the distance from the camera to the object in hand.

8. Collectable Objects

Children of BP_Master_CollectableObject_Inspect or copies of BP_CollectableObject_Inspect can be stored in the players inventory or put back in place when they are inspected. After placing the actor in your level, you will find various options in the details panel. Under the point "Setup" you can change all important data about an inventory object.

Inspect Distance	Distance between player camera and object
Simulate Physics	Use this toggle to enable physics simulation for collectable objects
Object Name	Displayed name of the object (e.g. "Black Mug.")
Object Description	Description text that will be displayed in inventory
Pickup Sound	Sound to play when the object is picked up
Object Image	Image to display in inventory *(Currently not in use, kept it for implementing a quick inventory in a future update)*
Is Key?	Can this object be used to open doors?
Key ID	If object is key, it opens all doors with the same ID
Discardable?	Can this object be thrown away?
Discard when Used?	Fetch this value inside a use action to decide whether the object should be removed from inventory after using it or not.
Use Action	To make an object usable, you need to assign a use script class to it. An actor of this class will be spawned then.

9. Use Actions

Use actions are actors of a specific class that can be assigned to inventory items. If an item is used the use action is spawned and will call its event (for example compare key IDs to open a door). The easiest way to create your own use action is to duplicate BP_UseAction_Key and replace the existing "Event Use Event" by your own use event. After the "Event Use Event" node place your code and then don't forget to destroy the actor afterwards.

Please have a look at the example use action to see how you can use the "Discard when Used?" value.

10. Inspection Objects with Hidden Clues

Since version 2.0 there are master blueprints for objects you can inspect and find clues in. For this you can find BP_Master_InspectableObject_Clue & BP_Master_CollectableObject_Clue. The first one is meant to be inspected without option to add it to your inventory, the second one can be inspected, put back or added to inventory.

In both blueprints, you can use an additional object, players can find (use the inherited "Clue" static mesh component for this). Or you can place the inherited box collision called "ClueTrigger" at the spot of interest you have on your object.

Focus on Clue Rotation	The orientation the object should rotate to in order to show the clue. You need to set this up manually by trying out values. I couldn't find a working automatic solution to this, but I assume this is fine.
Voice Over Sound	A sound cue that's played as the clue has been found. Leave this empty if you want.
Voice Over Delay	The time between the object has been found and the Voice Over Sound starts playing.
Added Item	This is the inventory item structure for collectable objects. If you want to add an object to the players inventory after the clue has been found, fill in the information about the object here. Leave this empty otherwise.
Journal Entry	Add an entry here, that'll be added to the players journal after the clue has been found. Leave this empty if you don't want to use it.
Clue Description	This text will be added to the object description once the clue has been found.

11. Readable Letters

Readable letters also have an Inspect Distance variable. To prevent shadows on letters while in hand, you may need to create a second material for your letter which you then set to unlit. Then set both materials for a specific letter actor in the Setup category within the details panel.

You can also display the letters content as plain UI text using the BP_Master_ReadableLetter_HUD blueprint as master. Place your text in the respective fields inside the Setup panel. To add a new line, press Shift + Enter while editing the text variable.

Since template version 1.5, you can choose another master blueprint for letters, called BP_Master_ReadableLetter_Font. This blueprint allows you to display the letter's content as user interface text.

12. Menu, Inventory, Journal

Since template version 2.1 the menu, inventory and journal are put together into one in-game-menu. This menu opens in a separate level with an own pawn (BP_MenuPawn). This way objects in inventory can't glitch through other actors in the game world as it would happen with the inventory immediately in front of the player.

To use this menu, go to Window->Levels and add MenuScene to your persistent level. By clicking on the little icon on the upper left corner you can change the levels details. Make sure it's not initially visible or loaded and set the translation far away from your level so it won't interfere with it.

The game controller will then load and unload it to create the menu, inventory or journal and also switch the Menu Pawn and back.

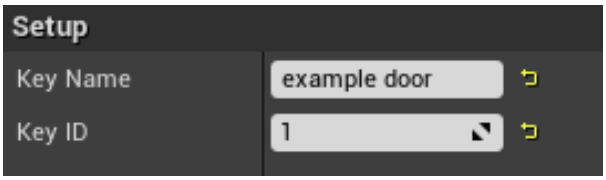
13. Journal and Collectable Entries

To create a journal entry, use the BP_Master_CollectableJournalEntry as master blueprint. Every child actor of this blueprint will be added to the inventory (stored within BP_GameController) and can then be read in the journal. The journal menu can be opened by pressing J on the keyboard or the left D-Pad up on the gamepad.

Like the readable letters, journal entries can have content over multiple paragraphs. There's no limitation by the user interface, since it is scrollable. A journal entry must have a valid name and headline. All information is set within the Setup category in the details panel.

14. Doors and Keys

Every child of all master door blueprints has a Setup category in the details panel. There you can define if a door is locked, assign a Key ID or define if a door should be open wide as default. Every key, created as child from the BP_Master_Key blueprint also has a Setup category containing a Key ID variable. A key with the ID 1 will open every door in the game with the same Key ID if collected. The variable Key Name will be displayed in the HUD as the key is picked up *(e.g. Picked up example door key).* This text can easily be changed within the BP_Master_Key blueprint.



When setting up your door actor make sure the door object's relative rotation Z is zero, otherwise it wont work.

With the new update, doors have a new variable called Select Key From Inventory. Enabling this option in combination with Is Locked? will open the inventory to search for the correct key when player interact with that door.

15. Cupboards and Drawers

Cupboards and drawers can have as many interactable objects (drawers and doors) as you need. You don't have to go into code for this! After creating a respective child actor, add a container mesh, all drawers or doors you need and put them in place. Make sure everything is facing the right direction without changing the relevant relative transform:

Drawers: **Relative Location Y must be Zero.**
Doors: **Relative Rotation Z must be Zero.**

You only need one mesh for cupboard doors. The one (right door) is placed regularly, the other door (left door) must be flipped on the x-axis! This is essential, so the system knows in which direction the door opens.

Drawer & Cupboard Combinations

If you have an object with doors and drawers, create a child from BP_Master_CupboardDrawerCombo or duplicate BP_Example_CupboardDrawerCombo. The blueprint will decide the correct action depending on the component tags "Door" and "Drawer".

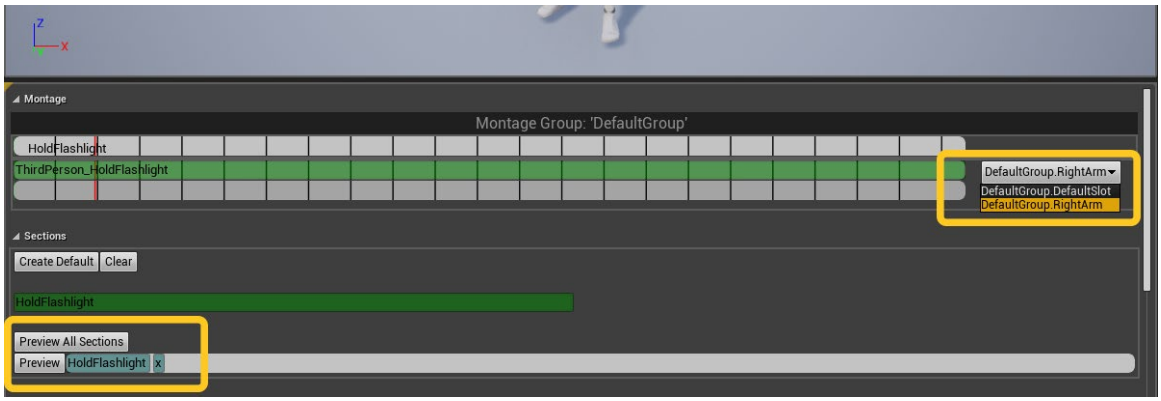
16. Drag Input

There are also master blueprints that allow players to "drag" doors and drawers as long as the button is pressed. These blueprints have another variable called Rotation Factor or Movement Factor. Changing this value allows you to control the translation from input to rotation or translation of the drawer or door.

17. Flashlight

To use the flashlight with your own character you need to make some changes and additions:

1. Create a new animation asset for holding the flashlight. Check the provided example animation for reference.
2. Create a new animation montage containing your new animation, create a new slot called "RightArm" and select this slot for your montage. Make sure your section is set to play a loop, see the screenshot below for reference.



3. Animation Blueprint: Change the event graph so it matches the provided example "ThirdPerson_AnimBP". (Right click on new copied variables and hit "Create Variable")
4. Animation Blueprint: Change your character's anim graph to match the provided example. If your skeleton hierarchy and naming differs from the mannequin skeleton, replace the bone names within following nodes to match your characters skeleton: Layered Blend per Bone & Transform (Modify) Bone.
5. Add a socket to your character's skeleton parented to its hand bone. Set your new created flashlight animation and add the flashlight as preview mesh to the socket. Transform the socket to place the flashlight correctly in your character's hand.
6. Correct your animation and socket if the placement is incorrect / not perfect.
7. The flashlight's material has a parameter that controls its emissive value (to make to front of the flashlight glow). Keep this in mind when you change the material or textures of the flashlight or if you're going to exchange the mesh. The parameter value is set within BP_GameController.

Optional: Open BP_FlashlightInHand to adjust the light.

18. Save and Load

The template contains a function to save and load its data (player location, inventory, door states, etc.) so you don't have to find out these respective variables yourself. If you want to add your own variables to the same saving and loading process, open BP_SaveGame first (Core folder). Add the variables you want to save here.

Next, open BP_GameController. There you'll find two functions, LoadGame and SaveGame. Open the SaveGame function and set the variables you want to save within BP_SaveGame before the "Save Game To Slot" node (Use the variable SaveGameInstance).

To load your variables, open the LoadGame function and drag all variables you need out of SaveGameInstance before the loading process ends (commented out).

19. Level Streaming

Please note: *If you're not sure what level streaming is and how it's used, please have a look at the official UE4 documentation first: https://docs.unrealengine.com/en-US/Engine/LevelStreaming/index.html*

There are a few things to be aware of in order to use streamed in sublevels together with this template, since we must manage collected objects, journals, etc. while we stream in or out parts of the game world. First, add all your sublevel names to the enumeration "StreamingLevels". This makes it easier to keep track of all your sublevels and allows an easy selection of levels you want to stream in or out.

Then open BP_GameController and look for the array variable called "Sublevels". Add all sublevels of your main level as elements to this array. You can of course also make this variable *Instance Editable* and change the array content on the instance inside your level, if you have multiple persistent levels with different sublevels assigned to them.

Now, the game will stream in (but hide) all sublevels automatically when the game starts. To stream in a level, please use the integrated trigger actors called BP_LevelStreamingTrigger or use it as reference if you want to stream levels in a different way. The important part is to use the functions inside BP_GameController, instead of the default streaming nodes. This way the game controller stores which levels are loaded when you save or load the game.

Another important thing to be aware of is that all sublevels of a main level must be loaded at all time but can be invisible of course. This is necessary to handle interactive objects in streamed levels.

If you're having questions about how to use level streaming with this template, please have a look at the example level or check out the Discord channel for further help.

20. Example Content

The audio files within this template are created by Tomasz Kostkiewicz and serve as simple example files for audio integration. However, you can use these in your projects if you want.

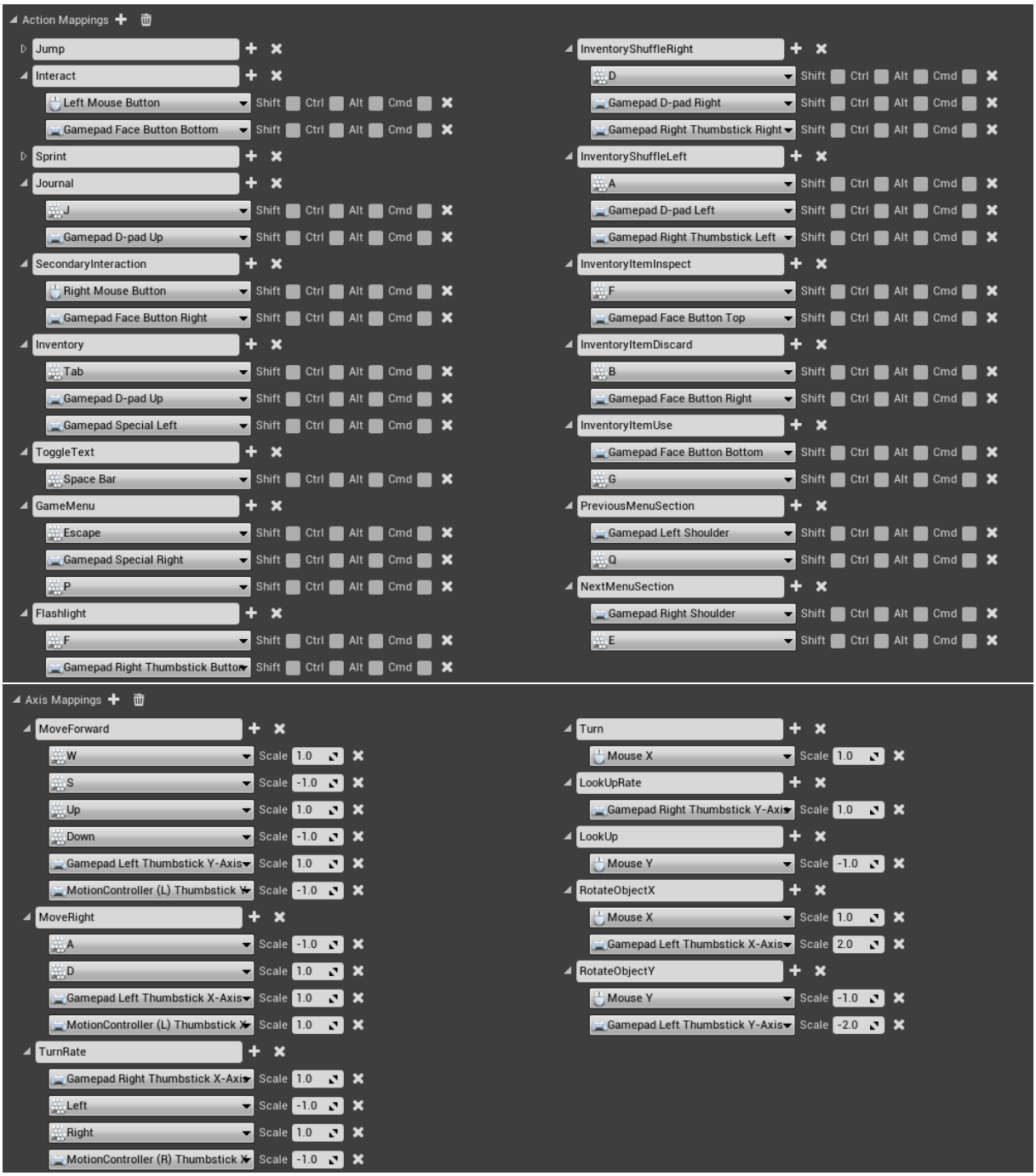
Mesh and textures are either engine demo content or created by me for demonstrational purposes. Feel free to use those in your projects with suitable attribution.

21. How to Migrate

The template can also easily be migrated in existing projects. To do this, follow these simple steps:

1. Create a temporary project from the downloaded template.
2. Migrate all files (main folder) into your project.
3. Set up the needed input. You can find the default input in the screenshot beneath this list
4. Place BP_GameController in your persistent level.
5. To use the save and load function, open your projects settings, go to Maps and Modes and replace the default game instance with BP_GameInstance.
    5.1. If you already use another game instance, add a variable (boolean) to your active game instance called "LoadFromSlot?". Then in BP_GameController at the top (the comment says "Load Game from Slot") there is a cast to node. Delete this node and create a new cast to your active game instance. From this pull out the "LoadFromSlot?" variable. Then also

within the Game Controller there are two functions: "PreLoad" and "LoadGame". Open these functions and change the cast node there, too.



22. Changelog & Update Help

To update the template to the newest version, I recommend creating a new project using the latest template version and migrating all changed files **separately** to your existing project. Keep the migration target project closed when migrating. When migrating one file, the engine also tries to migrate other files as well. To prevent a mess, just hit "Replace" for the file you want to migrate and "No All" for all other files. Use the following list to see which files have changed and need to be migrated.

Changelogs can be found on Discord: ***https://discord.gg/ekhzfCW***