Controls Manual

for World Builder VR

This manual contains all relevant information about the app’s control setup. You get to understand what the current controls are, how to change them and how to add new inputs.

Obsah

[1. Controls Overview 3](#_Toc90727566)

[1. Explore Mode 3](#_Toc90727567)

[2. Placement Mode 3](#_Toc90727568)

[2. Change Controls 4](#_Toc90727569)

[3. Add New Inputs 5](#_Toc90727570)

[1. Input Actions 5](#_Toc90727571)

[2. Code Structure 5](#_Toc90727572)

[3. Adding to the Code 6](#_Toc90727573)

[a) Adding Input to existing Modes 6](#_Toc90727574)

[b) Adding new input Modes 8](#_Toc90727575)

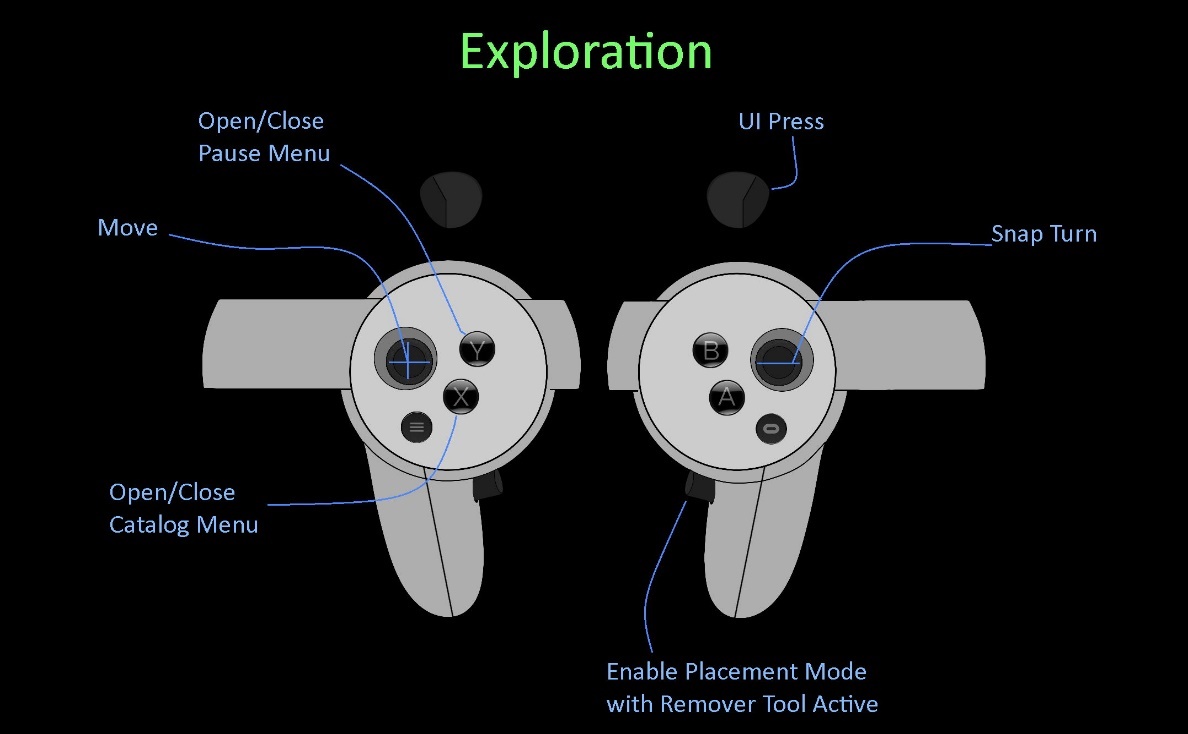
# Controls Overview

Here we describe how to control the app as of version dev1.01. Please pay attention to the fact, that this app was not tested with any other headset other than the Oculus Quest 2, so everything may not work as expected with other controllers.

There are 2 control styles used in the Edit Mode of World Builder VR: Explore & Placement.

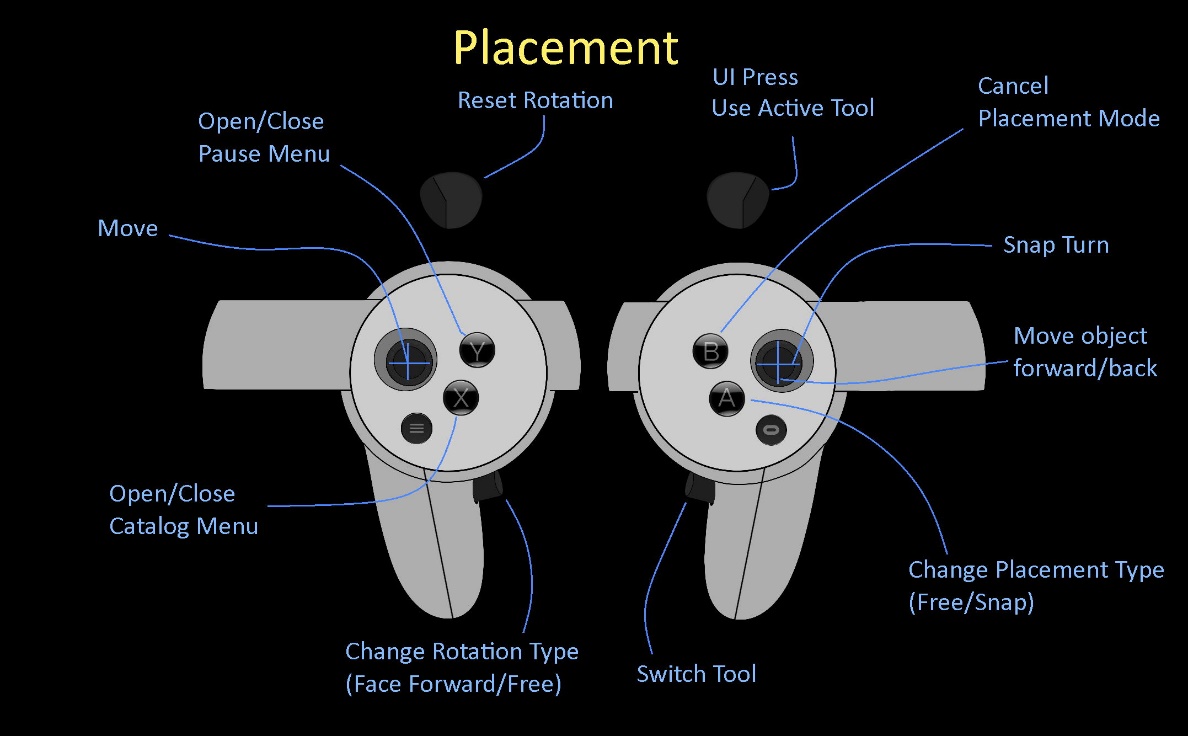
## Explore Mode

Explore Mode is activated when you enter the World Editor and when you disable Placement mode.



## Placement Mode

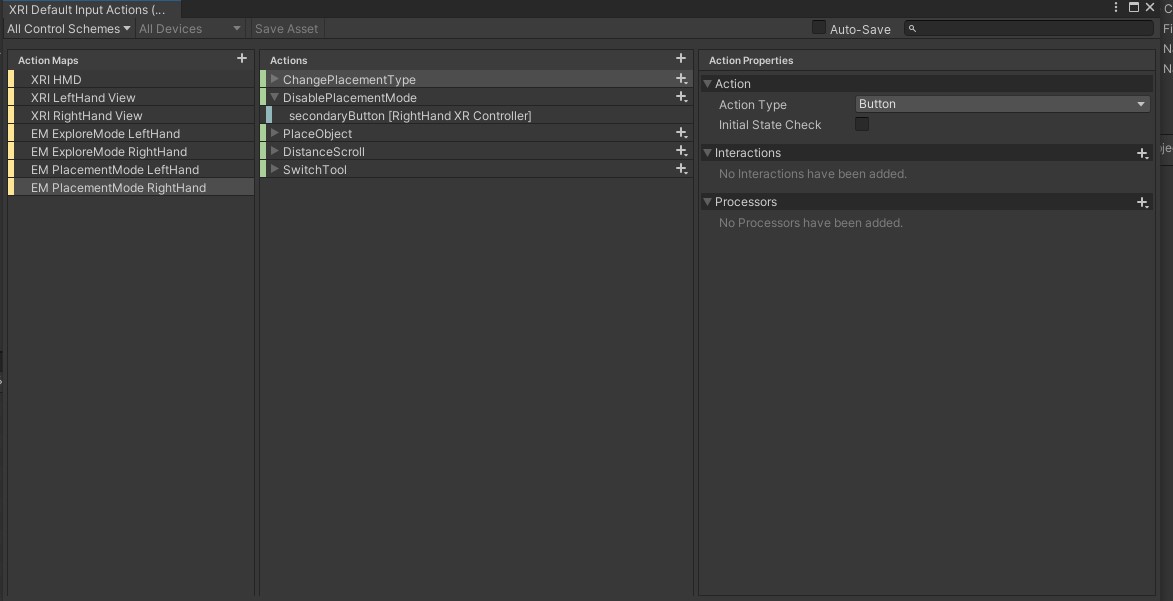
Placement Mode is activated by selecting a model in the Catalog or enabling the mode with a button press.



# Change Controls

If the current set of controls is satisfactory, all of them can of course changed. When you have the project opened in Unity, just go to:

Assets\World Builder VR\XR\Samples\XR Interaction Toolkit\1.0.0-pre.8\Default Input Actions\

and open the XRI Default Input Actions.

The app uses the first 3 Actions maps as base for all actions.

* EM Explore Mode contains all input actions for the Explore Mode (separately for each hand)
* EM Placement Mode contains all input actions for the Placement Mode (again separately for each hand)

All you need to do is select the Action Map you want to edit, then select the Action and there you can either change the assigned button or add a new one.

Unity’s New Input System is used throughout the app, so in case your are not familiar with very well, here is a [link](https://www.youtube.com/watch?v=Yjee_e4fICc&list=PLM0X88KflH_DMEx6HEYi4gugtPFLngAmQ&index=6&t=1422s) to a great overview video of the system.

# Add New Inputs

If you ever want/need to add custom inputs into the app, that twill do your own actions, this section is for you!

## Input Actions

To begin, add your new action into any of the existing Actions Maps. If you plan to make a brand-new section/mode for the app (for example a Model Builder), create a new Action Map for that mode, ideally for each hand.

Obsah obrázku text

Popis byl vytvořen automaticky

(Example, when we wanted to add the Placement Mode into the app, we created a new Action for each of hands called the EM Placement Mode and added all the relevant input actions in it.)

## Code Structure

Next is registering the actions into the app itself. For that to happen, I will describe the current implementation a bit. All relevant scripts for the input Systems are located:

Assets/World Builder VR/Scripts/Runtime/Systems/Input System

All under the namespace InputSystem, there are basically 3 types of scripts, that you should be interested in:

* Input Overseer
* Sectors
* Registries

**InputOverseer** – Sets up all and creates all relevant parts of the Input System. Also handles switching between Modes (for example, when we enable the placement mode, we want to disable the Explore Mode Input and Enable the Placement one, we do that here).

**InputSectors** – One for each Mode. These enable and disable Action Maps and send out events based on the inputs.

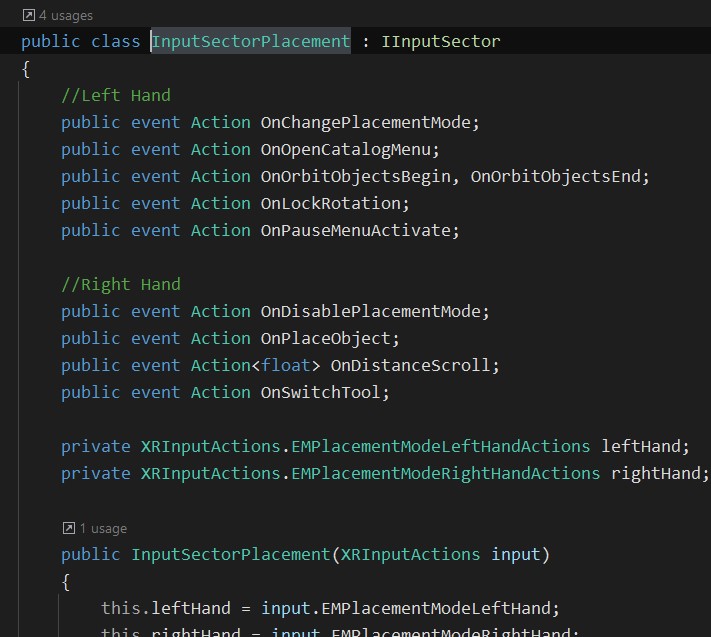
**InputRegistries** – One for each Mode. These are MonoBehaviours and their purpose is registering methods to input events read from the Sectors.

## Adding to the Code

### Adding Input to existing Modes

To plug newly added Actions into the application, first, go to the sector of the given mode. (For example, if a new type of action was added into EM Placement Mode Left Hand, go into the InputSectorPlacement.)

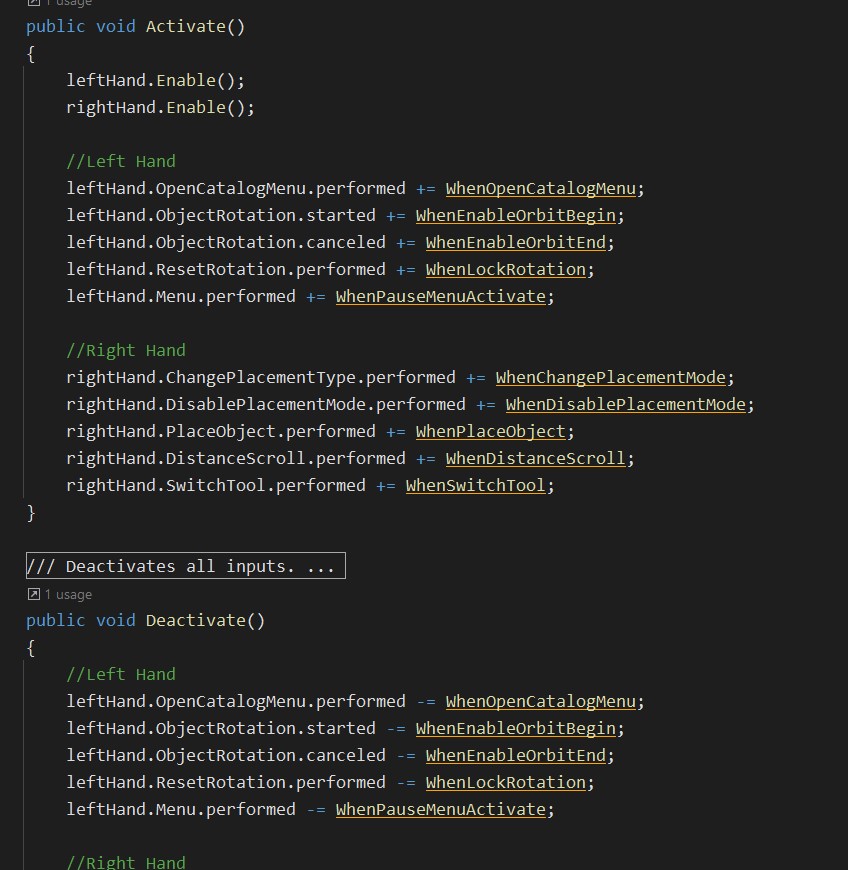
In there, create a new Action event named after the action type.



Then at the bottom of the script create a new function, that will raise the event.

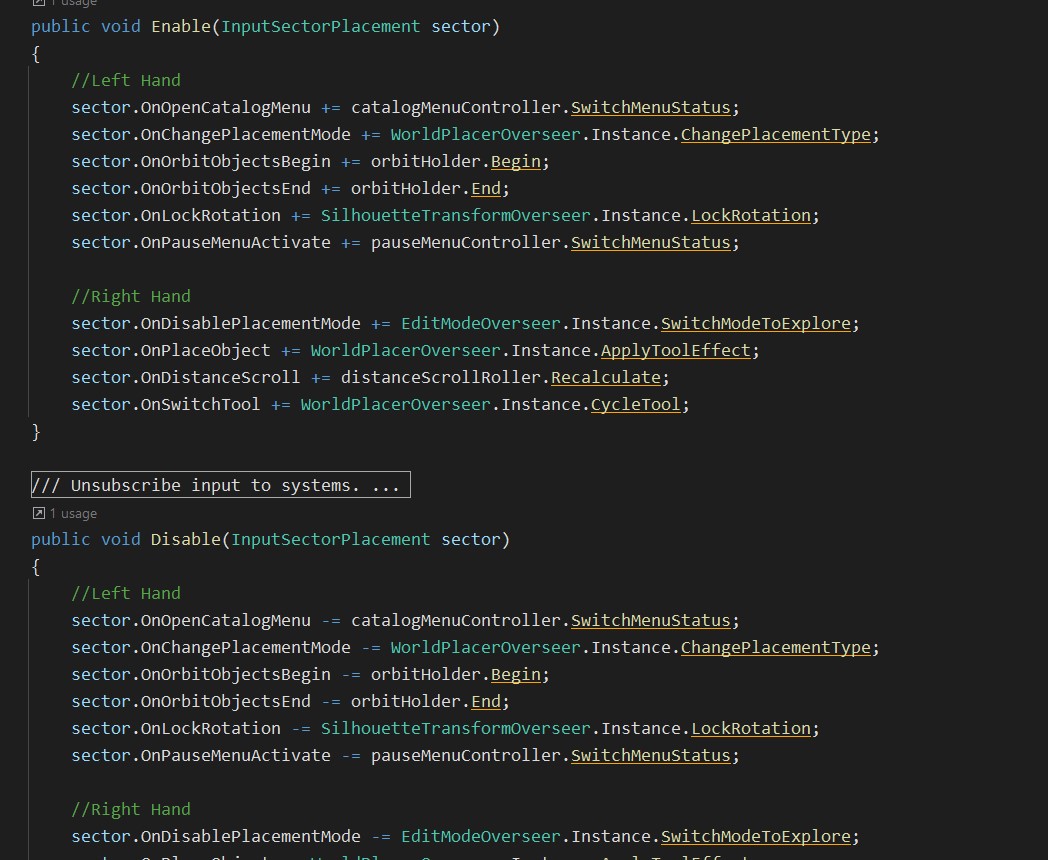


Finally, in the InputSector, register (and unregister) this newly created method under the specific event this Action fires when performed.



The event will now be fired when the Input Action is triggered. It is done this way to keep the Input System in a separate package and not drag the CallbackContext into other methods.

To register a method of your choice to the input event, go to the InputRegistry and register (also unregister) it under the sector event there.



### Adding new input Modes

Adding onto the Input System new modes is not that different, compared to just adding inputs to the existing segments/registries.

(Example when we wanted to create the object placing mode, we wouldn’t want to store its inputs in the Explore Mode, as some inputs would overlap, and it would not be considered very clean code.)

After creating your 2 Actions maps (for each hand) in the Input Actions, now you need to create a new InputSegment and InputRegistry in the same vein as the already existing ones.

Once you have a Segment and Registry, don’t forget to plug them into the InputOverseer (again do it in a similar fashion as the other existing ones.)