How to Add New Objects

# Introduction

This document explains the steps needed to add a new object in the game. There are several levels of adding a new object and each is explained in detail below. This document should convince anyone who tries this that we need to improve this greatly.

# Expose For Sim Use

This is the minimal level of exposing a new object. It will show in the picker UI, be place-able in the world, serialized to the level stuff file, and loaded from the stuff file. This includes a few helper routines that may or may not be used for programming support latter.

Note that adding a Prop is different than adding an Actor. The Props were simplified greatly but still require many steps.

## Adding a Prop

MicMill was the last to modify this code, although SCoy is the expert, there are pieces that have changed since it was created.

### Add the Model

Under the Content/Models filter, add the fbx file of your model to the project. Make sure the Content Processor is set to “UIModelProcessor”.

### Create your Lightweight Render Object class

Create a file under the SimWorld folder with a unique but simple name of your class ending it with LRO. Like GateLRO.cs.

Find the file CoinLRO.cs under the SimWorld folder. Copy the contents of this file into your file and replace all instances of CoinLRO in the file with your class name (GateLRO).

Update the constant string that defines which model to load to list the name of your model.

### Instance Creation

Find and Edit BokuGame.Cs.

In the LoadGraphicsConent() method, find the line “BokuGame.Load(CoinLRO.GetInstance());” and duplicate it, and replace CoinLRO with the name of your LRO.

In the UnloadGraphicsContent() method, find the line “BokuGame.Unload(CoinLRO.GetInstance());” and duplicate it, and replace the CoinLRO with the name of your LRO.

### Save, Load, and Paste

Find and edit the file SimWorld\XmlLevelData.cs. Find the ReadFromXml method, duplicate the case statement that has CoinLRO.GetInstance() call and modify it to match your class. For the case statement use a unique simple lower case string (this same name will get referenced in the DropGateAtCursor as a parameter to the GameProp constructor. For Props, the WriteToXml is handled in a generic way so no other changes are needed here.

Find and edit the file Scenes\InGame\InGameEditObject.cs. Search for the line that contains "Boku.SimWorld.CoinLRO". You should see an IF block of code that is duplicated many times; duplicate an instance and modify to clone your class (rename as needed).

### Object Add Menu

Find and edit the file Scenes\InGame\InGameEditObjectAddItem.cs.

Search for the DropCoinAtCursor method, duplicate it and modify it for your class. The string name should be the same referenced in the previous “Save, Load, and Paste” section. The Physicalities parameter is used to define if the object can be picked up. Look at others for examples.

Find and edit the method AddUiSelector. This is where the menu items are added to the Object pie menu. You can search for CoinLRO to see the example of how the coin is added to the menu. Duplicate the section and place it where it makes sense to add to the menu or sub-group.

Find and edit the method SelectNewItemSelector. Find CoinLRO case statement and duplicate it and have it call your DropObjectAtCursor method you created above.

## Adding an Actor

SCoy is the expert here and should be referenced as needed.

### Add the Model

Under the Content/Models filter, add the fbx file of your model to the project. Make sure the Content Processor is set to “UIModelProcessor”.

### Create your Object classES

Find the file HoverCarLRO.cs under the SimWorld folder. Copy the contents of this file into your LRO file and replace all instances of HoverCarLRO in the file with your class name (GateLRO). Update the constant string that defines which model to load to list the name of your model. Review all the other GameActor LRO files for rendering techniques and modify to match your need.

Find the file SimWorld\HoverCarMenuItem.cs. Copy the contents of this file into your MenuItem file and fix it up.

Find the file SimWorld\HoverCar.cs. Copy the contents of this file into your class file and fix it up.

### Instance Creation

Find and Edit BokuGame.Cs.

In the LoadGraphicsConent() method, find the line “BokuGame.Load(HoverCarLRO.GetInstance());” and duplicate it, and replace HoverCarLRO with the name of your LRO.

In the UnloadGraphicsContent() method, find the line “BokuGame.Unload(HoverCarLRO.GetInstance());” and duplicate it, and replace the HoverCarLRO with the name of your LRO.

### Save, Load, and Paste

Find and edit the file SimWorld\XmlLevelData.cs. Find every instance that references HoverCar and duplicate and modify to your class.

Find and edit the file Scenes\InGame\InGameEditObject.cs. Find every instance that references HoverCar and duplicate and modify to your class.

### Object Add Menu

Find and edit the file Scenes\InGame\InGameEditObjectAddItem.cs. Find every instance that references HoverCar and duplicate and modify to your class.

# Add a Filter for it

These are the steps to add a filter that will show up in programming for your new Prop or Actor.

## Add a Icon Texture

Create a texture that is the icon of your Prop or Actor with no text. The name should follow the standard “filter.name.png”. Add this texture to Content\Textures\Tiles\ and make sure the Content Processor is set to Texture (Spirte, 32bpp) – XNA Framework.

## Add the Filter to the list

Find and edit the file Content\Xml\CardSpace.Xml. Find the section FilterPieces. You can find an example by looking for “filter.boku”. Duplicate the ClassificationFilter. The upid is a unique identifier for it; and normally should be changed to match your texture file. The label is the string that will be applied to the tile and shown to the user. The name property is the name that is used in the code changes above that is the unique simple lower case name.

The order of the filters is the order that they are displayed.

Another property “group” could be present that defines a display hint to include the item inside a group. The value is a reference the group containers in the GroupPieces section at the end of the file.

# Add a custom verb

These are the steps to add a custom verb for a GameActor.

## ADD the actuator to the list

Find and edit the file Content\Xml\CardSpace.Xml. Find the ActuatorPieces section. Add a line item in the list like the following. Note the order in the list is the order it will be shown to the user.

<VerbActuator upid="actuator.open" Verb="open" Valency="Intransitive" actuatorCategory="Action" mountkey="gate" label="open"/>

Upid – the unique identifier of the actuator; also used to look up the icon image

Verb – the verb from the Verb enum this tile represents

Valency – the valency of the verb (look it up). Intransitive means the verb only has a subject.

actuatorCategory – category of this verb, in this example a general action

mountkey – list of names of objects this verb can be exposed from, only needed if there is a restriction, in this case the gate is the only one supporting open.

Label – the text to display with the icon on the tile.

## Add the Verb

Find and edit the file Base\GameThing.cs. Find the Verbs enum and add the Open verb.

## Add Support for the Verb

In your actor class, find the DoDirectObjectVerb method, and include a case statement for the GameThing.Verbs.Open. Within it implement the code for the verb. For a simple example of this look at HoverCar.cs. Look at the support of GameThing.Verbs.NoiseOn.