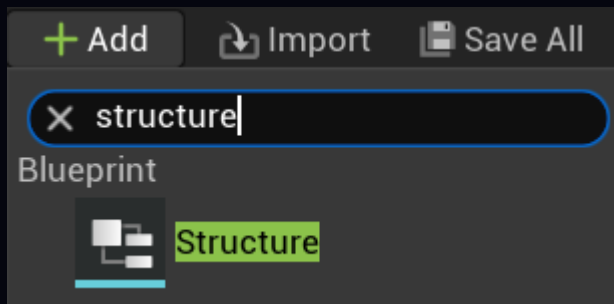




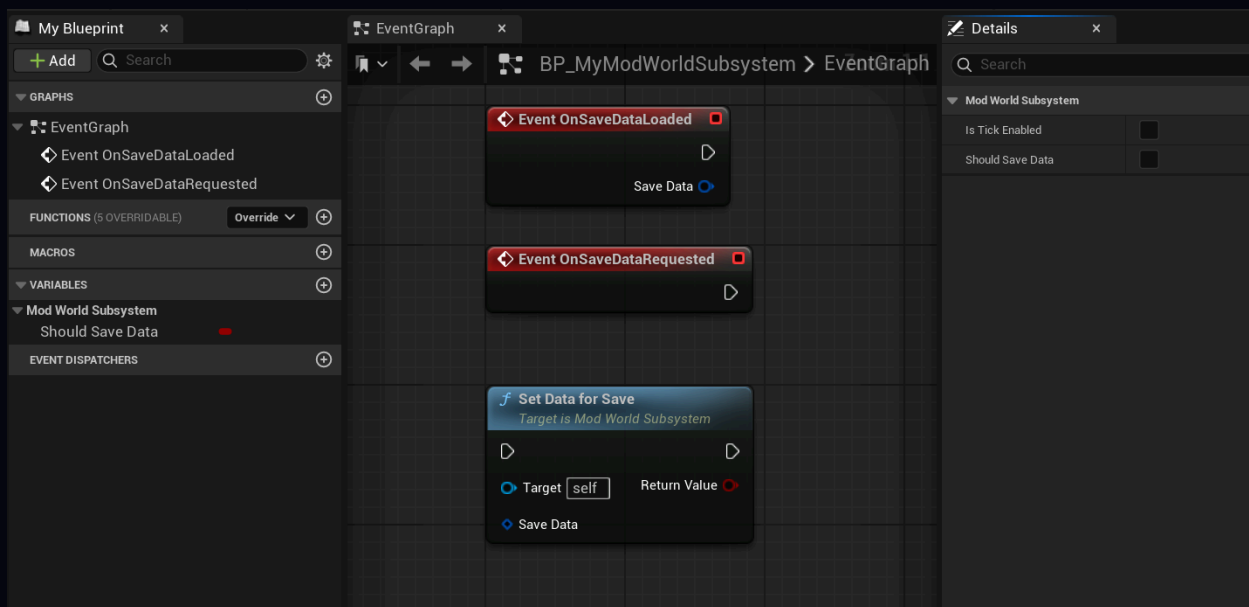
# SAVE/LOAD SYSTEM FOR MODS

## Overview

You can save and load data needed for your mods. The system uses UE's property serialization and lets you save properties of any type available in the editor. It works with Structure Assets, which you can create and configure as you normally do in UE.



Your starting point is *ModWorldSubsystem*. Here you can find the bool property *Should Save Data* and several events and functions related to the system: *OnSaveDataLoaded*, *OnSaveDataRequested*, *SetDataForSave*.



## ShouldSaveData

Determines whether your *ModWorldSubsystem* should participate in the Save/Load system. By default, it is set to **false**. Remember to turn it on if you need it.

## OnSaveDataLoaded

This event is called by the system shortly after loading a save slot when your data is ready. It provides an Instanced Structure object as the SaveData arg for you to then unwrap it into a struct object of your Structure Asset type.

## OnSaveDataRequested

This event is called by the system during saving, suggesting that you pass the data you would like to place in a save slot. It does not save anything on its own, it only notifies you about the moment when the system is ready to receive your data.

## SetDataForSave

This function lets you set your data that will be placed in a save slot next to other default game data. It returns true when the data meets restrictions and false otherwise.

# Restrictions

You can save up to 10 MB of data in one save slot, including some overhead from UE's property serialization.

## Example

As an example, let's take a look at a mod adding a basketball hoop that tracks how many grenades the player throws in it. It contains the following assets:

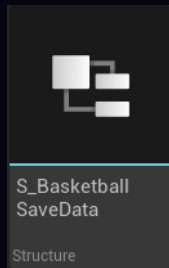
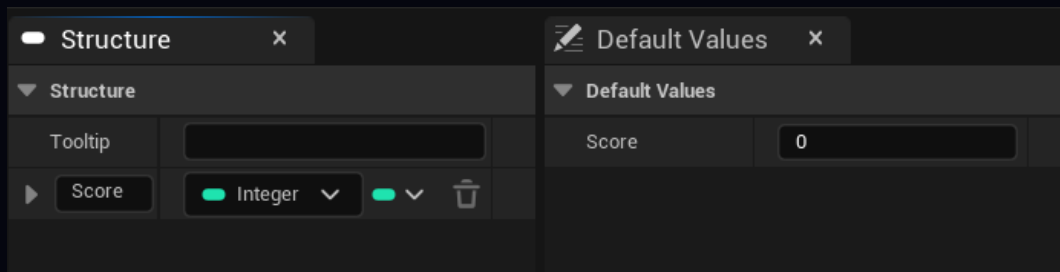
- *BP\_BasketballHoop* - blueprint actor of the hoop itself.
- *BP\_BasketballWorldSubsystem* - *ModWorldSubsystem*, which works with the Save/Load system.

- *S\_BasketballSaveData* – *Structure Asset* representing data that is going to be saved and loaded.

*BP\_BasketballHoop* will not be covered as it is outside the scope of the guide.

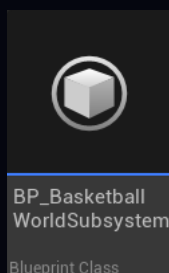
### S\_BasketballSaveData

It consists of just one property: **Score** (type – **Integer**). This is what you need to pass to the Save/Load system to remember the player's score.



### BP\_BasketballWorldSubsystem

It uses everything discussed above to save and load the player's score data.



My Blueprint

+ Add

Search

GRAPHS

EventGraph

Event OnWorldBeginPlay

Event OnSaveDataLoaded

Event OnSaveDataRequested

FUNCTIONS (4 OVERRIDABLE)

Override

MACROS

VARIABLES

Mod World Subsystem

Should Save Data

BasketballHoop

LoadedData

BP Basketball Hoop

S Basketball Save Data

EVENT DISPATCHERS

Details

Search

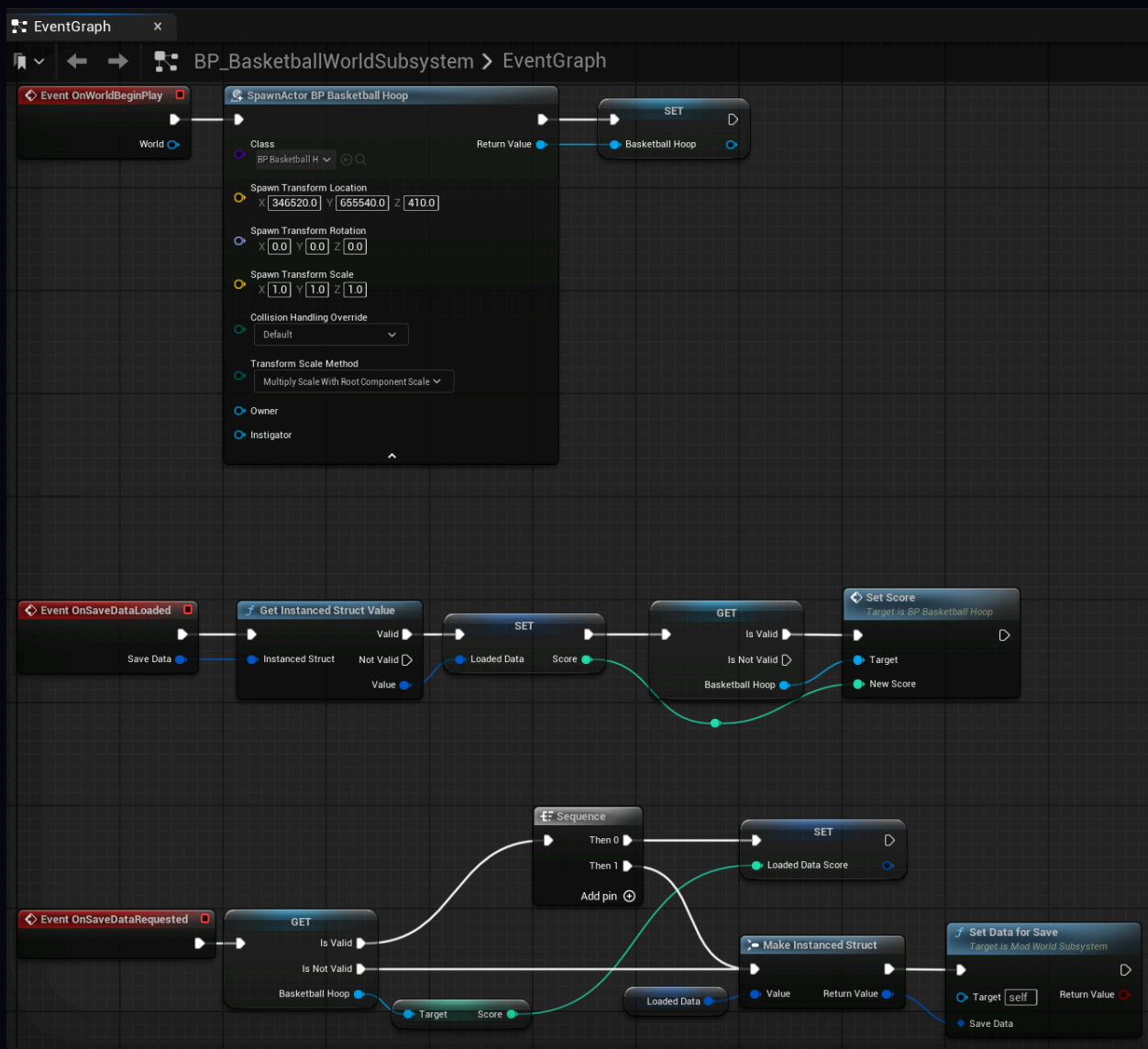
Default

Loaded Data

Mod World Subsystem

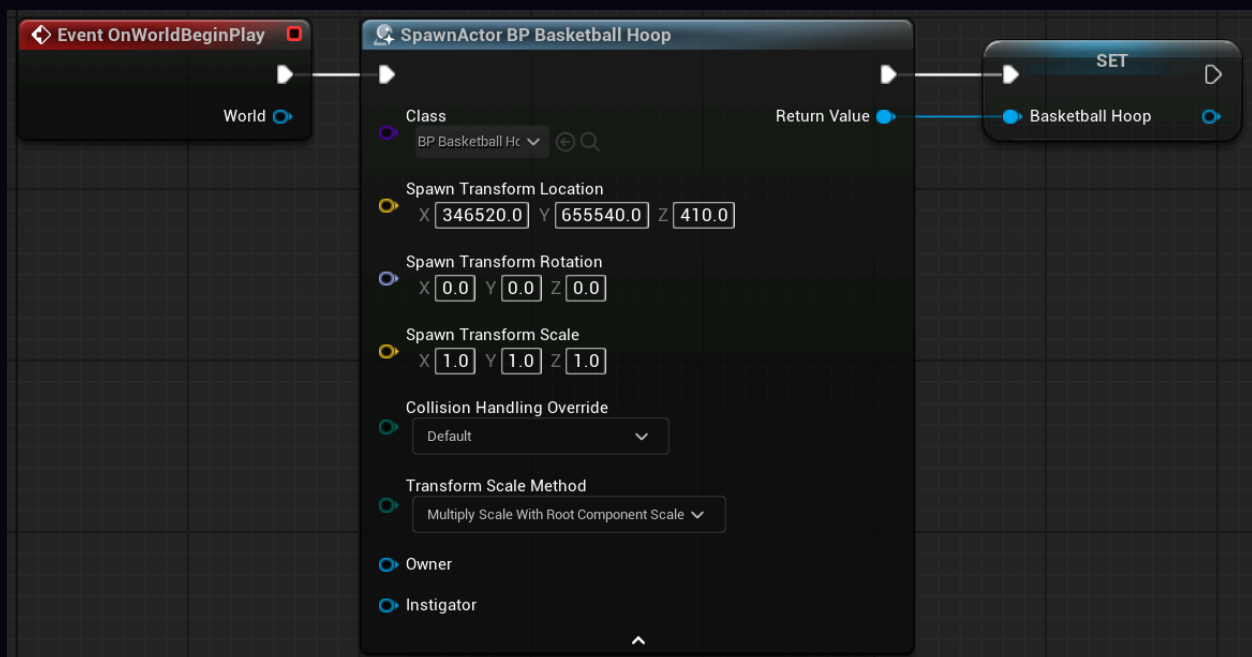
Is Tick Enabled

Should Save Data



## OnWorldBeginPlay

Spawns an actor of *BP\_BasketballHoop* and caches it in the *BasketballHoop* var.



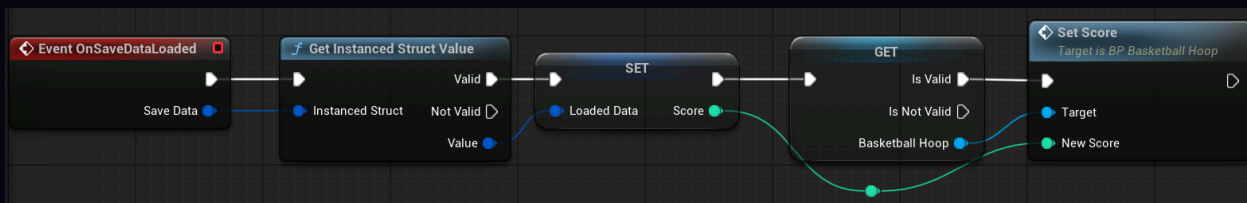
## OnSaveDataLoaded

- Unwraps the Instanced Structure object received as the SaveData arg from the event using the *GetInstancedStructValue* function.
- Sets the result in the local var called LoadedData (type – *S\_BasketballSaveData*) and, for further convenient use, immediately breaks the struct from the right pin into a list of containing properties (in this case, it is just Score).

### NOTE

You can break structs by right-clicking on the corresponding pin and selecting Split Struct Pin.

- Gets the cached BasketballHoop object, checks it for validity (right-click on the getter node of *BasketballHoop – Convert to Validated Get*), and passes Score from *LoadedData* to it by calling the *SetScore* function defined in *BP\_BasketballHoop* – in turn, it updates visuals on the actor.

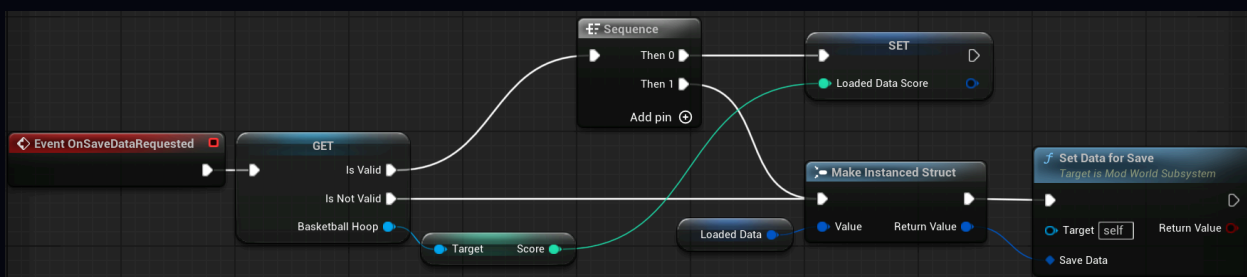


## OnSaveDataRequested + SetDataForSave

- Gets the cached *BasketballHoop* object, checks it for validity:
  - If valid, updates Score inside LoadedData with the actual Score tracked by *BasketballHoop*.
- Converts *LoadedData* into Instanced Struct using the *MakeInstancedStruct* function.
- Passes created Instanced Struct to *SetDataForSave*.

### NOTE

The returned value of *SetDataForSave* is ignored for simplicity, but you might want to take it into account while working on your mods.



Now, the mod is able to add the player's score to a save slot and restore it when it is loaded.

