b mod creation How-To, documentation and reference

b mod creation intro.mp4

https://www.youtube.com/watch?v=QfyM31rvpPo

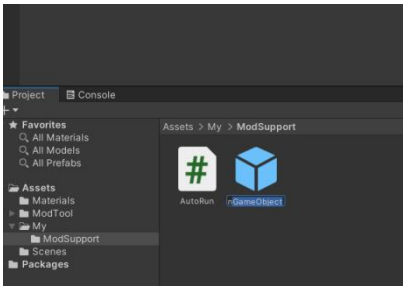
Required (exactly): Unity 2022.1.6f1

Project Type: 3D built-in renderer

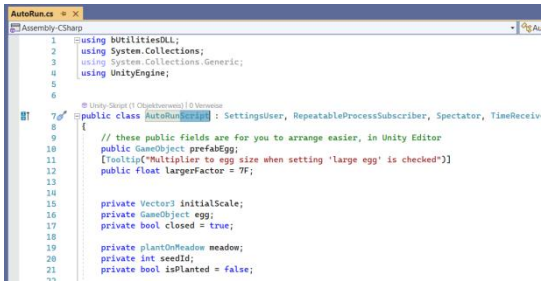
Import Package „b-mod-tools.unitypackage“ from the game folder in the Steam Library

Export by choosing „Tools -> ModTool -> Export Mod“ in the menubar and giving a name and a location

Entry Point: Rename „Assets/My/ModSupport/AutoRunScript“ to „AutoRun“,



afterwards open it and change the class name equally.



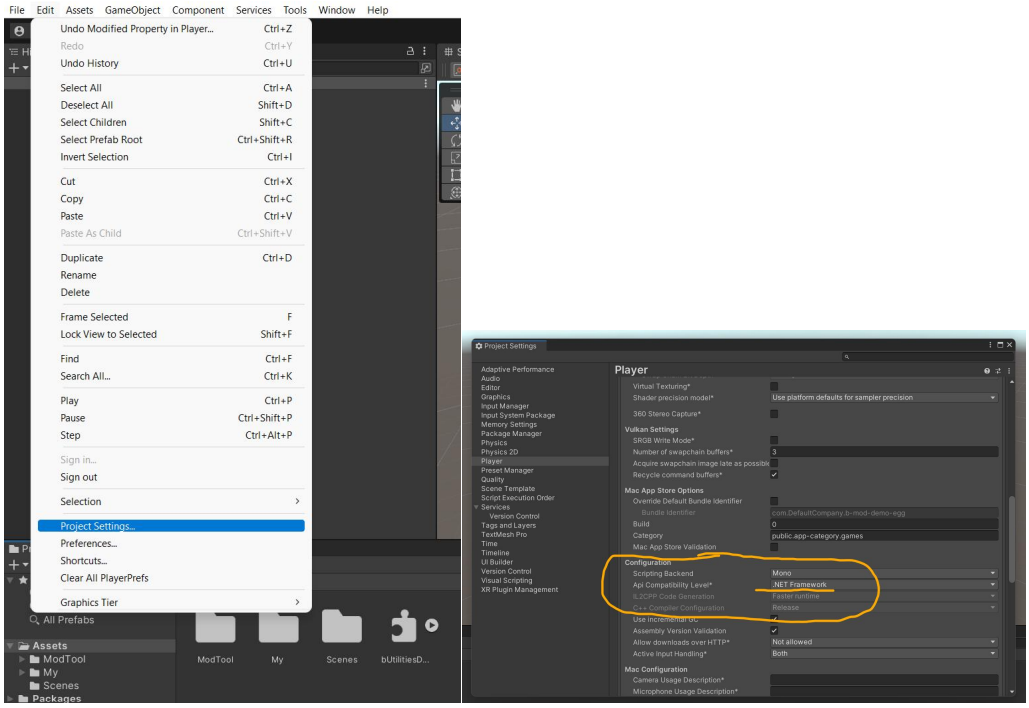
Rename „Assets/My/ModSupport/AutoRunGameObject“ to „AutoRun“ as well.

The „AutoRun“ prefab is the only prefab which gets instantiated when the mod loaded and its attached „AutoRun“ script run.
Ensue all your work from „AutoRun“. The scene is not being exported to the mod but you can use it to test. New prefabs and scripts of yours get exported to the mod.

The „AutoRun“ script itself contains all major interoperability functionality and relevant considerations as a working demo.
You are free with the exception of System.IO and System.Reflection.

Following functionality is available additionally:

requires „using bUtilitiesDLL;“



Change the Api Compatibility Level **to „**.NET Framework**“ in the** Project Settings **before you let** export **your mod !**

Helper

name	returns	arguments
`getB`	the GameObject being b	-
`getMeadow`	the plantOnMeadow component, see that page	-
`createUid`	an application-wide unique string	-
`getMinusYReference`	the Vector3 representing the point „far below“	-
`nop`	nothing, does nothing, suitable for passing as callback when you don‘t require one	-
`nopA<T>`	nothing, does nothing, suitable for passing as callback which takes argument of type T when you don‘t require such	1: type T, is ignored, range: any

TimeTaker

name	returns	arguments
`takeTimefor`	-	1: interface `TimeReceiver`, you need to implement `void timeTransitionsToNight (bool itDoes)` which is called when day transitions to night and night to day
`stopTakingTimefor`	-	1: interface `TimeReceiver`, same object you passed to `takeTimefor`
`getDayCompletionFromDawn`	a float from 0F to 1F linearly interpolating from dawn to dawn across day and night, .5F is day end / night start	-

BLift

name	returns	arguments
`inform`	-	1: interface `Spectator`, you need to implement `void bTransitions(bool intoSpace)` which is called when b transitions to and when b transitions from space
`stopInforming`	-	1: interface `Spectator`, same object you passed to `inform`
`getDistanceFromHorizon`	a float being the distance from the horizon where b would transition into or out of space	-

plantOnMeadow

name	returns	arguments
`plantSeed`	the id of the seed planted, as an int	1: GameObject being the prefab to plant 2: int being the amount of argument 1 to plant 3: bool when true indicating to treat argument 2 as the density per 10,000, 1 meadow edge part is 10, the meadow has an equal number of parts around 1 central part, meadow edge/2 length == 5 is 11 parts resulting in 12,100 4: string, when not empty being the name of a setting, see Settings, which contains the value to use as amount instead of argument 2 5: Vector3 being the target to aim at when planting which is done from above and thus ought be a point „far below“ (argument 1 is planted on the meadow below which this arguments value should be) 6: float offsetting the plant in the vertical direction when planted 7: bool when true

		indicating the meadow is a sphere, argument 5 should then be the center of that sphere 8: Action<GameObject[]> to call when planting occurred passing all planted instantiations of argument 1 to it
`exchangeSeed`	-	1: `int`, being the id from `plantSeed` 2: `GameObject`, being a prefab to replace argument 1 to `plantSeed` which yielded the return of the id passed as argument 1 here
`growBy`	-	1: `int`, being the id from `plantSeed`, lets grow the GameObject passed as argument 1 to `plantSeed` which yielded the return of that id
`getBigTree`	the GameObject being the central big tree	-

Settings

allows you to add controls to the settings menu opened by pressing the Escape key on the keyboard

AutoRunScript.cs is a SettingsUser which gets notified in OnApply when settings have changed. You can then let get the values of the controls you added from the protected property settings, of type Settings.

You can let create slider and checkbox.

name	returns	arguments
`createSlider`	bool which is true when creation is succesful	1: string which shall uniquely identify the setting 2: columnPosition, an enum which can be left or right 3: string which is the label 4: float which is the initial value 5: float which is the minimum 6: float which is the maximum 7: bool which when true lets the slider be in integer steps from argument 5 to argument 6 8: string which is the help text 9: Color of argument 3 10: Color of argument 8
`createCheckbox`	bool which is true when creation is succesful	1: string which shall uniquely identify the setting 2: columnPosition, an enum which can be left or right 3: string which is the label 4: float which is the initial value 5: string which is the help text 6: Color of argument 3 7: Color of argument 5
`getValue`	a float being	1: string identifying the

	the curent value represented by the control identified by the argument 1 passed here, either 0F or 1F for checkbox representing unchecked and checked	setting to get the value of 2: optional float being the value returned when the setting to be identified by argument 1 is not found

FormContinuer

seamless continuation of a sine wave of frequency A at a point by a sine wave of a different frequency

name	returns	arguments
`FormContinuer`	- (the object, is constructor)	1: float being a time offset for when the original wave does not start at time == 0F 2: float being frequency A as factor to time progress of the wave
`continueForm`	float being the value of the wave at argument 1	1: float indicating the transition time 2: float being the different frequency **This setup does appear to not fullfill the desire before the creation of this functionality, investigation to do**