Bridge : Serials

FOREWORD

In contrast with the other WordPress assets in this series, Bridge : Serials require virtually no work for you to do in Unity at all and does most of what it does in the WordPress dashboard

To get started with the kit, just install it into your project and, if you haven't already done so, upload and activate the wub_serials plugin on your WordPress website. You only need to install the plugin once and you are all set for all current and future games.

ABOUT Bridge : Serials

So what is Bridge : Serials exactly? Simply put, it allows you to pre-generate any number of serials you want and then it let's players take ownership of them by linking their user accounts to a serial.

Going a little fast there? Let me slow it down. Every user, once they register on your website, gets a unique ID. When they register with a valid serial that ID is linked to that serial. This means 4 very important things:

1. Nobody else can use that serial any more because It is already allocated to someone

2. That serial is linked to that game for that person regardless of the device he is playing on. This means that he can register the game once on one device and then the game is registered on all his devices automatically. This includes any future device replacements or even when borrowing someone else's device for a while. The serial is linked to THEM, not to one of their devices.

3. As inferred from the aforementioned point, this means the serial is only ever needed once to register one copy of the game on one device. After that the serial never needs to be entered anywhere by the owner ever again. Not even when they contact customer support since you already get to see all the serials for all their games just by calling up their account in the dashboard. This means you don't have to display the serial in the game unless you want to and for all sense and purposes the owner can completely forget about the serial after he used it.

4. This also means that "sharing" the game or selling it second hand makes absolutely no sense. The game cannot be played without a serial and the serial is linked to a user's account. If they ever decide to "share" the game then they would have to give the person access to their entire website account including the ability to change their username, email address and password. If they want to sell the game second hand they first have to contact you and ask you to remove the serial from their account. Now the other party still needs a license (which you can allocate to them with one click) but if the original owner ever wants to play the game again he will have to buy another serial for himself again. Piracy of your game(s) no longer make any sense…

Bridge : Serials comes with a complete dashboard interface for generating and removing serials. You can reserve serials for games being distributed in retail stores and (if you have the
Bridge : Money plugin installed) you can also sell your games directly via your website. This causes a non-reserved serial to be allocated and registered to the buyer automatically meaning they never have to enter the serial even once. They don't ever even need to see the serial as everything will just work from the moment your WooCommerce transaction completes.

Also on the dashboard are displays listing all available and all allocated serials, per game. Also a means of viewing all serials belonging to a single user. When viewing a user's account you can simply select a game from a drop down box and, with one click, register a serial to him for that game. Alternatively, next to every serial he owns is a button to remove that person's registration for that particular game.

Absolute control over all things registration related in no more than 2 clicks for any action…

MUCH ADO ABOUT NOTHING

From within Unity Bridge : Serials only exposes 3 functions for you to call:

1. **RegisterSerial**
   If a customer owns a serial and has not yet registered the game, use this function to send the serial to the server to allocate that serial to that user.

2. **ValidateRegistration**
   Use this any time you wish to check if a game is registered to the currently logged in player. It takes no parameters and will either return a bool named "valid" with a value of true or it will trigger your error response function. This is useful at the start of the game or if you just want to inject random checks to minimize any kind of game hacks

3. **FetchSerialNumber**
   This also takes no parameters. Just be sure the user is logged in when you call it and when it's done you can use the results as you see fit. **NOTE:** The serial is NOT needed for anything after the first registration so the results of this function are purely for you to use in some cosmetic way

SEAMLESS INTEGRATION WITH THE LOGIN PREFABS

I have modified the stock login prefabs to automatically detect if you have Bridge : Serials installed or not. If you do it will expose a couple of new fields for you to set in the inspector:

- **Bool Check_for_serial**
  If true, the login process will include the validation of a the user's registration and set the static bool WULogin.HasSerial accordingly

- **Bool Fetch_serial**
  If true the registered serial will be returned for you to display in game should you wish to do so (or whatever else you want to do with it)

- **Bool Require_serial_for_login**
  When true it changes the behavior of the Bridge's Login prefab. Instead of closing the prefab so players can start to play, it now redirects to the screen where they have to either register with a serial they have in their possession or they can click on the button to buy a serial if they do not already have one. Additionally, the PostLoginMenu screen's "Continue playing" button now also validates that a legit serial exists for the user.

  NOTE: If Fetch_serial was set to true then these fields will be handled for you automatically. If you had them on false then you will have to make a manual call to the server using the ValidateRegistration function mentioned above.

- **String Product_url**
  If this field is set then clicking on the button to buy a serial will take the player to this url

NEW STATIC PROPERTIES AT YOUR DISPOSAL

The Login prefabs now contain the following new static properties:

- **Bool RequireSerialForLogin**
  The private variable you set in the inspector was meant for the login function to know whether or

not to check for the serials. For the rest of the game this variable is what you will check to determine if a serial is required to play the game or not

- **Bool HasSerial**
  As the name implies this checks to see if a valid serial has been found for the user. The login prefabs set this value on login if you elected to have it check for serials but if you call ValidateRegistration then you will need to manually update this value to reflect the results you received

- **String SerialNumber**
  This property is not guaranteed to hold any value and in most cases is not needed. The login prefabs set this value if you had Fetch_serial set to true during login. If you call FetchSerialNumber manually you need to populate this property with the results.
  This exists purely for cosmetic reasons, nothing else

NOTES

Some thoughts on what you can do with this. If you enable the Require_serial_for_login option then you are effectively forcing people to pay before they can play. This means you can make your game available for free download and collect your money afterward.

As an alternative, you can have the login function check for a serial during login but *NOT* have it require the serial for login. By doing this you can allow people to play the first few levels of your game and at some point you can add a test to see if WULogin.HasSerial is true. If not you can (for example…) (A) pop up the login prefab again and show them the screen where they need to enter their serial or (B) instead of loading scene *n*, just load a scene that says "*Thank you for playing the demo. If you want to continue playing, please purchase a serial number now*" …and with that you have turned this kit into a means for making demo games from any game.

You can very easily redo the validation test again at any point in the game (on game start, on an enemy dying, on saving the game, on using a health pack, before loading any new scene, etc.)
Just check "Does this player have a serial?". Trust the local bool or fetch the results from the server again. Your choice

WARNING

As you will see in the modified demo, adding this kit to your project requires a slight change in the way you approach logins. Before, when a login was successful you could just start to play the game. Now you also have to take into account whether or not the serial is required.

The OnLoggedIn event still fires after the login completed so if your custom code just waits for that event and then loads the new scene you will have to modify that code to return, instead, if WULogin.HasSerial is false. Instead of registering to WULogin.OnLoggedIn register to the OnResumeGame event instead and check at that time whether or not you can continue the game.

It's a minor change but an important one. See the updated demo for more

LINKS

For more and for up to date info on any updates or new features please feel free to visit the products page at
http://mybadstudios.com/product/bridge-serials/