Extensible XP System

Documentation - v1.0.0

The Extensible XP System aims for a simple XP system that you can expand upon. Quickly change XP formulas and how the XP should be calculated.

How to use

NOTE: For versions v5.6.7f1 or v2017.4.30f1 the prefabs does not work because of Unity's new prefab workflow introduced in v2018.3. Code runs fine though. If you are running these versions just skip importing the prefabs (they are very simple prefabs, see images below).

NOTE: If you are only after the XP system then everything you need is in the /Scripts/XP folder. And if you don't want to use RuntimeSets: find and remove the canReceiveXPSet references from XPReceiver.cs file.

1. Import the package.
2. Included in the package are default ScriptableObjects (in the /Data folder) that you can use if you want.
    a. Otherwise you create these using the project menu (right-click inside the Project view -> Create -> Saucy -> Modules -> XP -> ...)
3. Add the script components XPGranter.cs and XPReceiver.cs to GameObjects you wish to Grant/Receive XP.
    a. Assign XP grant method to the XP Granter GameObject.
    b. Assign XP receive method to the XP Receive GameObject.
4. Hook up XPGranter.GrantXP() so it gets called.
5. Use XPReceiver's UnityEvents to listen for updates.
6. Done, have fun.

What is included

/Data

ScriptableObject assets that act like data containers for: XP formulas, XP grant methods, and XP receive methods (more on those below).

/Documentation

Self-explanatory.

/Prefabs

- **UI prefabs**: Used for showing how the package works.
- **XP prefabs**: Simple GameObjects for having the XP script components which lives inside the game (MonoBehaviours).

/Scenes

Demo scene for showing how the package works.

/Scripts/RuntimeSets

ScriptableObjects that keeps a list of references so other MonoBehaviours can reference it on runtime. The lists can be used for anything you'd like, a common one is keeping a list of GameObjects (example included).

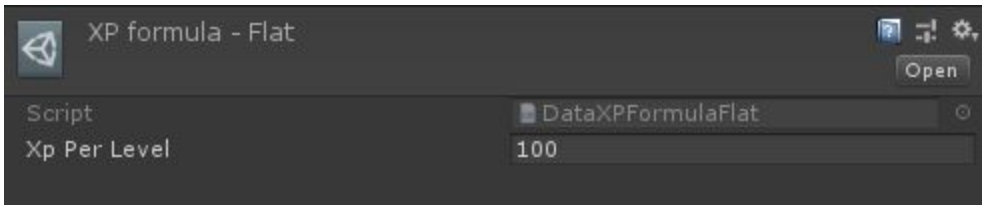
/Scripts/UI

Scripts that the UI uses for displaying XP. Not needed for the XP system to function.

/Scripts/XP/Interfaces

"Contracts" how XP must be granted and received.

/Scripts/XP/XP formulas

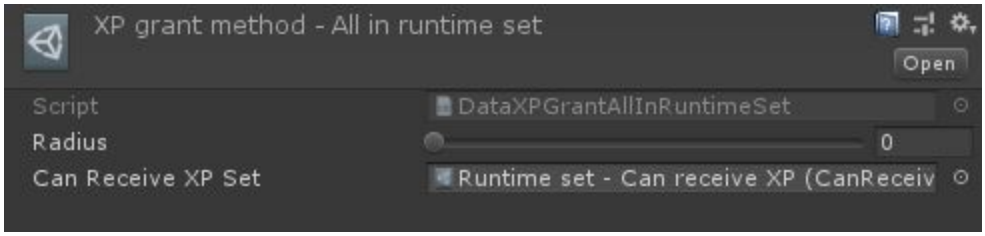


Contains all formulas on how XP is calculated. These scripts are ScriptableObjects that you create in your project and you (or your designers) can change the values in the inspector. You can expand and create your own formula, just inherit from DataXPFormula.

Xp Per Level: Base field that you can define however you'd like. In the receive methods I've created the required XP for the next level is increased by Xp Per Level (100).

/Scripts/XP/XP grants

ScriptableObjects that contain how XP should be granted. I've included two examples I think are the most used methods.

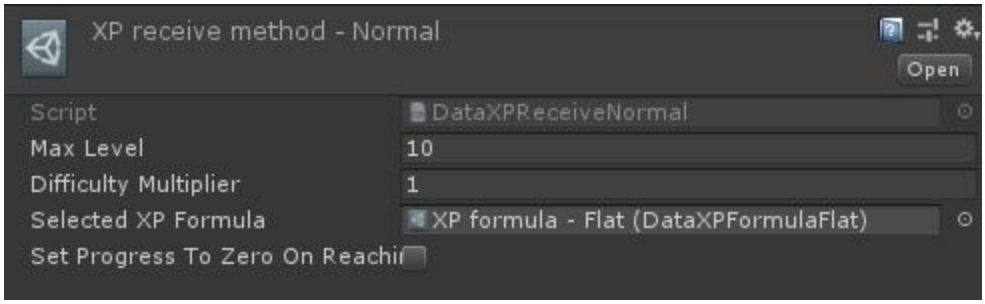


- **DataXPGrantAllInRuntimeSet**: Grants XP to all objects (in this example all players).
- **DataXPGrantInAnArea**: Checks everything in a sphere for objects that implements the IXPReceive interface. You can change the layers which the script should look for IXPReceiver interfaces.

Radius: The radius how far from the GameObject that has XPGranter.cs will look for colliders with the IXPReceive interface (in this case we don't use it so we set it to 0. It is mainly used in **DataXPGrantInAnArea**, but you can use it for your own methods).
Can Receive XP Set: The runtime set we'll loop through and grant XP to the items in the list.
Layers To Check For Receive XP: Here you can specify which layers to check for the IXPReceive interface.

/Scripts/XP/XP receivers

NOTE: The ScriptableObject assets that are created from these scripts are the ones that store the XP.
These scripts do the heavy lifting. Based on the XP formula selected it calculates the current XP, current level, etc.



In these assets is also where assign the maximum level. A difficulty multiplier is included if you want to use it (higher value increases the XP gained).

I've included two XP receiving methods that I think are the most used XP system out there.

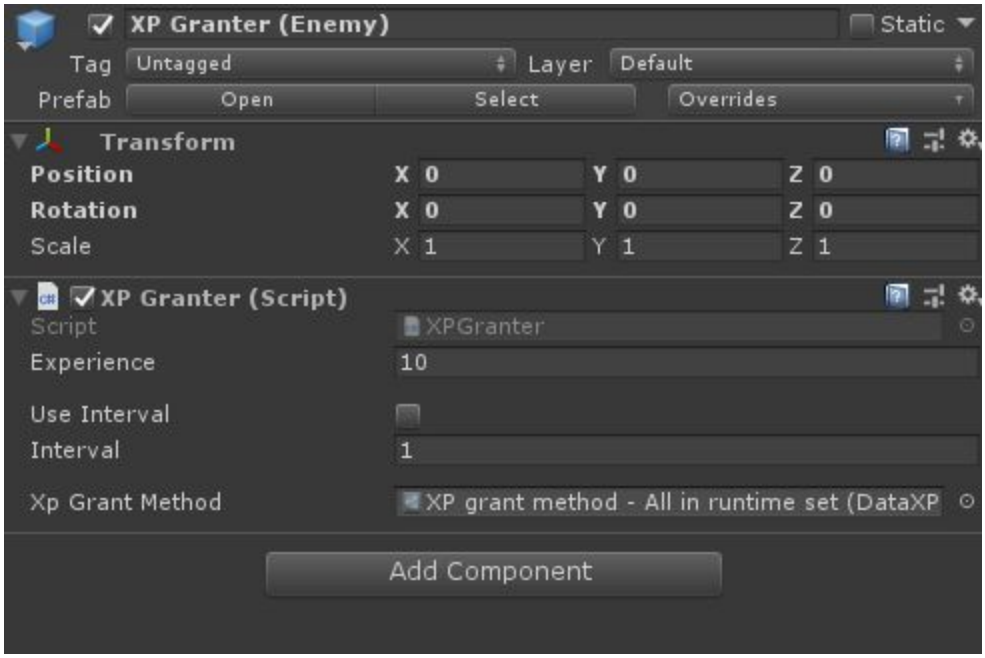
- **Normal method**: Works like World of Warcraft does their XP. It keeps keep all the XP you've gained and just increases the required XP for the next level. Does not reset the current XP.
- **Alternative method**: Works like Horizon: Zero Dawn does their XP. It resets current XP to zero after you gained a level and increases the required XP to level up.

Max Level: Maximum level, we can set this in the inspector.
Difficulty Multiplier: A multiplier we use when adding current XP and acquired XP.
Selected XP Formula: Which XP formula to base the calculations on.
Set Progress To Zero On Reaching Max Level: Set progress to zero upon hitting max level (similar to how World of Warcraft does it).

/Scripts/XP/XPGranter

The script that grants XP. Could be added to an enemy, and hooked up that when it dies it grants XP. Call XPGranter.GrantXP() to grant XP.

I've also included an Interval field that calls GrantXP() each specified interval (runs an IEnumerator, and the default value is OFF and 1 second).

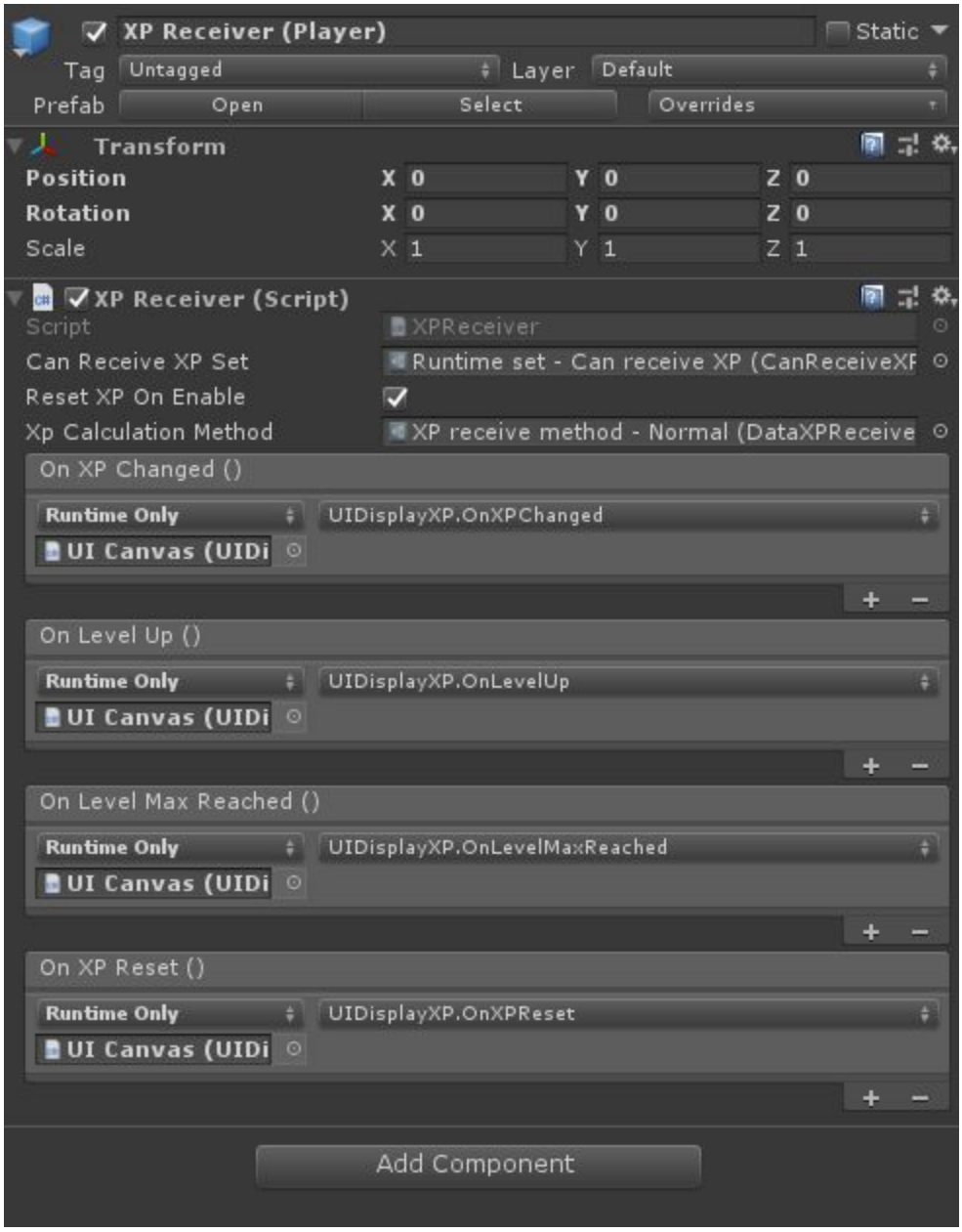


Experience: The amount of XP to grant the receiver.
Use Interval: Enabled grants XP each interval.
Interval: How often XP should be granted.
Xp Grant Method: Which XP Grant method that should be used when calling GrantXP(). This can easily be changed by just drag & dropping another XP Grant method in the inspector.

/Scripts/XP/XPReceiver

The script that receives XP. Could be added to a player, and depending on how you are granting XP this GameObject receives if the criteria is fulfilled.

Has UnityEvents that you easily can hook up to UI (for example to display something when the player levels up).



Can Receive XP Set: Reference to a RuntimeSet that this script adds itself to, so it can receive XP.
Reset XP On Enable: Reset the acquired XP back to zero on OnEnable.
Xp Calculation Method: The XP calculation method that is going to save the XP, calculate current XP and level, calculate missing XP, etc.

Below are UnityEvents that you can use to call things like the UI, other GameObjects to let them know XP has changed, the player has gained a level, etc.
On XP Changed (): When ever gained XP has been added to the acquired XP this event is called.
On Level Up (): When ever a new level is reached this event is called.
On Level Max Reached (): When maximum level is reached this event is called.
On XP Reset (): When ever the XP is reset this event is called.

Changelog

Version 1.0.0
- Initial release

Credits

- RuntimeSet created by Ryan Hipple for Unite 2017.
- Square XP formula created by aaaaaaaaaaaa.

Feedback

If you have any issues please create an issue on the project's repository (found here).

Suggestions or constructive criticism are appreciated. Pull requests for additional XP formulas, grant/receive methods is also welcomed.

Contact & Support

Email: support@saucy.se
Website: http://saucy.se/
Repository url: https://gitlab.com/Saucyminator/unity-package-extensible-xp-system
Issues: https://gitlab.com/Saucyminator/unity-package-extensible-xp-system/issues