



Unity Integration

Welcome to anonymous in-app purchases.

V 1.0

Content

1. How the plugin works
2. Step by step integration

How the plugin works

Important: Purchase is done through **Tonkeeper!**

Key points:

- The principle of IAP and its storage in a **publisher wallet** is based on the formation of an encrypted comment and check it when the user logs in.
- The encryption key is formed from **2 keys**: the **application key** and the **user password**. So, the encryption key will be unique for each user.
- The comment of the purchase (purchase id) is used as the encrypted information.
- While checking comments, the amount of payment is also checked. The convenience is that it is possible to pay from any wallet.
- The user just needs to know his username and password after purchase.

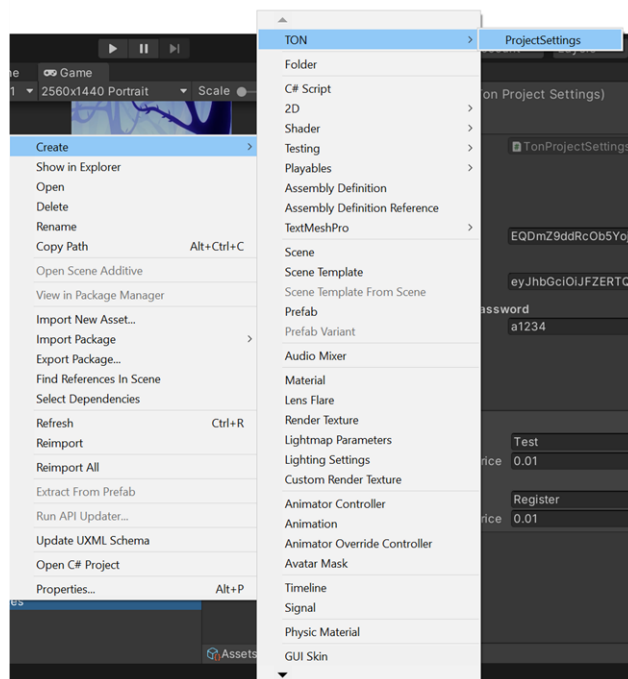
What is needed in the application when developing:

- Address of the The Open Network wallet.
<https://tonapi.io>
- The client key that is generated in the Telegram bot.
https://t.me/tonapi_bot
- To create a password for your app for encryption purchases comments. It is very important to keep this password to decrypt all purchases made.
- It is desirable to use a password of no more than 20 characters.

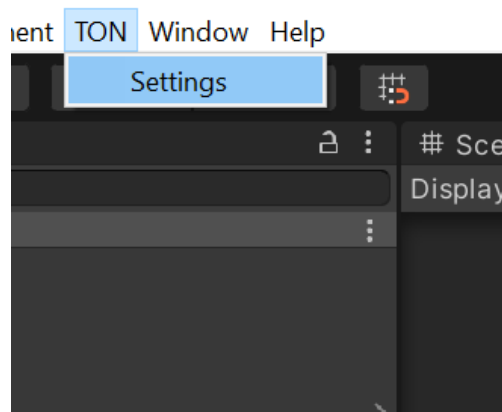
Step by step integration

1. After importing a package you must find or create Ton Settings (Scriptable Object) to store keys and purchase IDs.

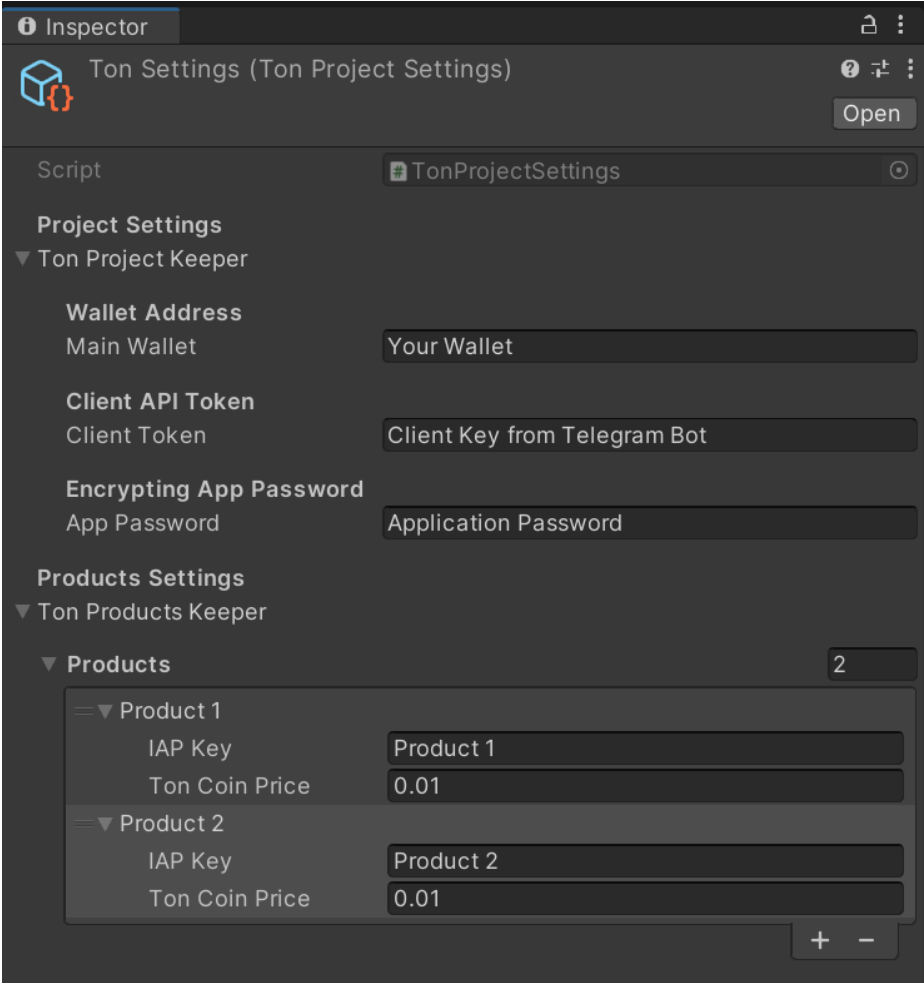
How to create new



Where to find the existing



2. You need to fill in the fields in the selected Scriptable Object. (There must be only one in the project)



The screenshot shows the Inspector window in a development environment, displaying the 'Ton Settings (Ton Project Settings)' script. The script is named 'TonProjectSettings'. The settings are organized into two main sections: 'Project Settings' and 'Products Settings'.

Project Settings

- Ton Project Keeper**
 - Wallet Address**
 - Main Wallet: Your Wallet
 - Client API Token**
 - Client Token: Client Key from Telegram Bot
 - Encrypting App Password**
 - App Password: Application Password

Products Settings

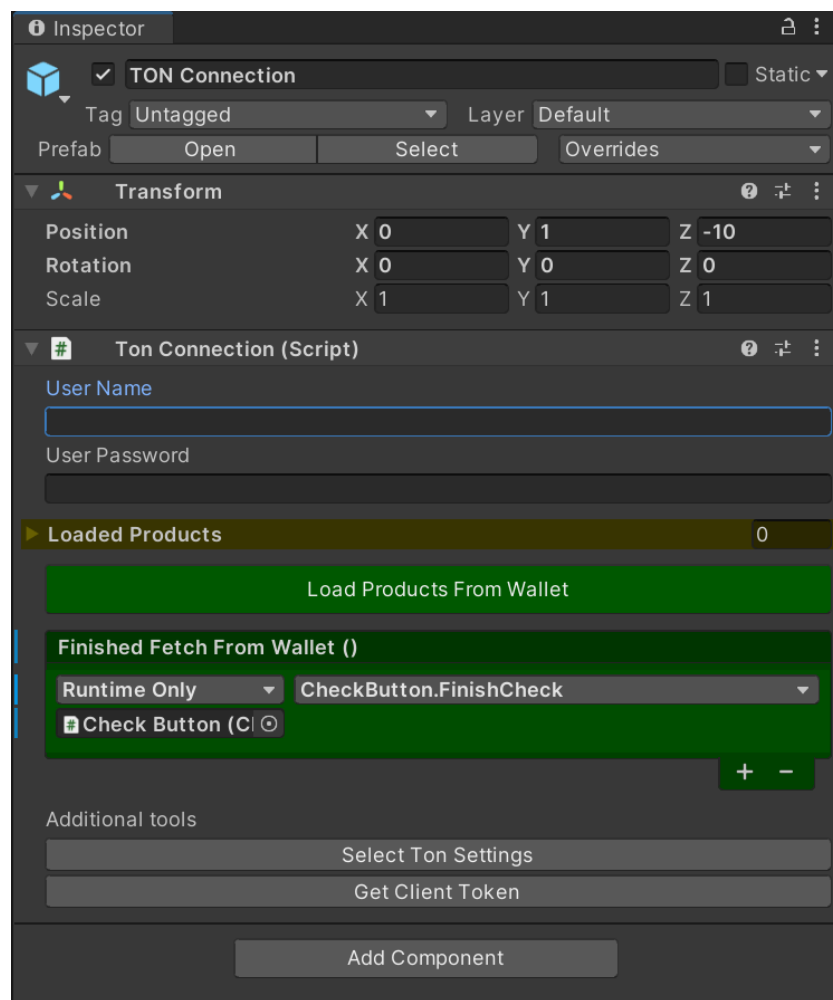
- Ton Products Keeper**
 - Products** (Count: 2)
 - Product 1**
 - IAP Key: Product 1
 - Ton Coin Price: 0.01
 - Product 2**
 - IAP Key: Product 2
 - Ton Coin Price: 0.01

At the bottom right of the Products list, there are '+' and '-' buttons to add or remove products.

You can create purchases with the same IDs, but with different prices.

Important. The test speed has not been tested on large applications with many users.

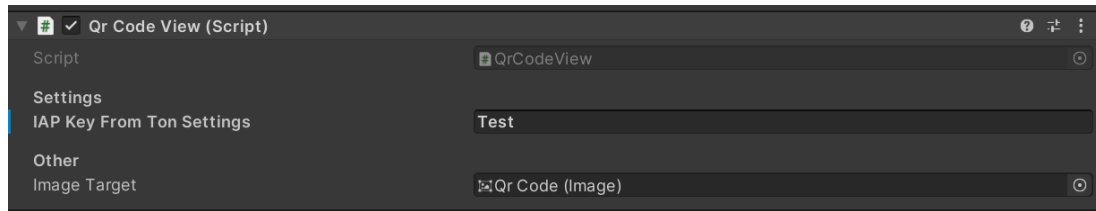
3. The next step is to work with the **Ton Connection** [Game Object](#). You can create it yourself or choose from the prefab. You can also use it in editor mode. To pull purchases from the wallet, you must enter a username and password. If they are correct as well as the application password when purchasing, the purchases made will be successfully uploaded to this game object. You will see them in the **Loaded Products** array. You can click the green [Load Products From Wallet](#) button or use the **Start Load Products From Wallet** method in the script to start loading your purchases. When the download is complete, the **Finished Fetch From Wallet** [Unity Event](#) will be called.



Then you can process your purchases yourself or see how it's done in the **Check Button** script in the **Demo**.

4. To create a QR code of new purchase on the screen you need to familiarize yourself with the [QR Code View prefab](#) and the **QrCodeView** script. For the easiest code generation, you need to enter an existing [purchase ID](#) in the inspector and call **the Find Local Product Key And Generate** method in the script.

In the [Image Target](#) field, place your UI element with the [Image](#) component to generate a sprite in it.



5. Check to see if your purchases are working.

Thanks for reading!

If you have any questions or suggestions, please email: andrew olenk@gmail.com.