

## Simple Crypto WebGL Templates (v1.3)

Thank you for purchasing my asset and I hope it can get you started developing your NFT integrated game.

### Contents

Introduction	1
Getting Started / Demo	1
Testing In-Editor	2
Workflow / Script References	2

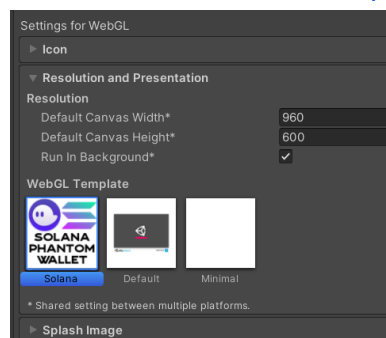
## Introduction

This asset is designed to give you a quick start to developing NFT based games by loading and adding the user's owned NFTs into an easily accessible list for you to use however you please. You can choose to use the list for the NFT's Public Address or use the created CryptoNFT ScriptableObject list that is created.

## Getting Started / Demo

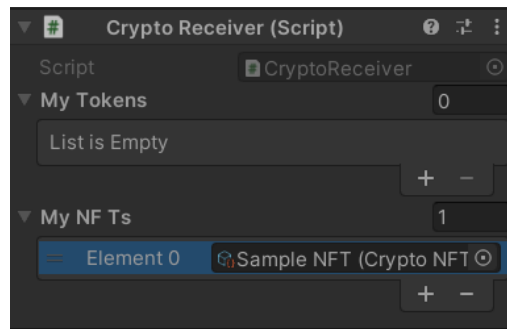
You can easily test the asset works by going to Project Settings and selecting one of our templates, then just hit 'Build & Run'. Unity will make the build and launch your browser. Simply wait till the WebGL has loaded and hit the 'Connect' button. Then test it by pressing the 'Update NFTs' button in the game and it should load a new card for each of your owned NFTs. In v1.3 I have also included a 3D Gallery Demo so you can choose which scene to include in your Build to test.

You can see a quick demonstration of the Card Demo at <https://youtu.be/uTJJsEFX0ig>



## Testing In-Editor

I have included a Sample NFT in the Prefabs folder that can be used but you can create your own demo NFT by right clicking in the Project area and going to 'Create > Puppy Gaming > NFT' and just filling in the Metadata yourself for the new NFT. Then when your game is running in the editor select the CryptoReceiver GameObject in the hierarchy (nested under DontDestroyOnLoad), add a space in the My NFTs list and drag your test NFT to it.



## Workflow / Script References

### [Puppy Gaming/Scripts/Plugins/solana.jslib](#)

This file contains all the code needed for the WebGL player and also the code needed to log into the wallet from the selected chain, then loading all the wallets NFTs and finally grabbing the metadata from them and sending them to the CryptoReceiver GameObject in your scene. The functions in this file are commented for easier understanding.

### [Puppy Gaming/Prefabs/CryptoReceiver.cs](#)

The CryptoReceiver prefab should be placed in your first Scene when the player would be logging into their wallet, it will persist across scenes so no need to add it anywhere else. The CryptoReceiver script receives the data from the Login request process and puts the metadata from each NFT into a CryptoNFT ScriptableObject and then instantiates that object in the MyNFTs List from the *ReceiveNFT()* function. This list can be accessed by any other script with *CryptoReceiver.CR.myNFTs*

The current accessible data from a CryptoNFT object is;

- .nftName - The NFTs name,
- .spritelink - The image URL for the NFT,
- .description - The collections description.
- (more metadata to be added)

It also creates a list of NFT Public Addresses that can be found in *CryptoReceiver.CR.myTokens* which helps if you don't need to grab metadata and just want to use the Addresses for comparisons.

You can get the user's Wallet Address from *CryptoReceiver.CR.walletAddress* and a shortened version of the address for in-game display like 'AaAa....BbBb' from *CryptoReceiver.CR.shortAddress*

You can check whether the address has been collected with *CryptoReceiver.CR.isConnected* bool

## Provided Demos

### Card Demo

This demo provides a sample of how to instantiate cards into your scene from the NFTPrefab objects in the *CryptoReceiver.CR.myNFTs* List.

Feel free to reuse any of the code from it or just edit the scene to your liking.

### Gallery Demo

The same applies from the Card Demo, but the NFTs will be instantiated as art works in an art 3D art gallery environment (apologies for the poor character controller)