

Settings Saver

Overview

Creating A Settings Saver

Available Settings Savers

- Player Prefs

 - Append Data

 - Player Prefs Key

- Json & Xml

 - Append Data

 - File Name

 - Directory Name

 - Directory Provider

 - Directory Type

 - Persistent Data Path

 - Data Path

 - Custom

 - Custom Directory Path

Writing a custom settings saver

- LoadSettings

- SaveSettings

- DeleteSave**

- AppendData

Overview

A settings saver object is used to save the current values for the settings. The settings saver is a scriptable object and needs to be referenced in your settings collection. If your settings collection has a reference to a settings saver it will use the settings saver's save and load implementations.

Settings savers for PlayerPrefs, Json and Xml are provided by default with the asset.

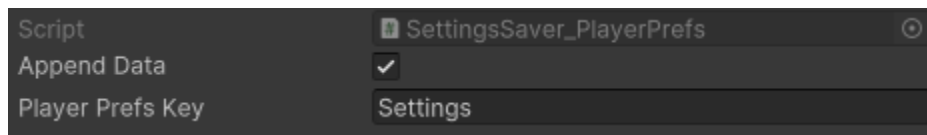
Creating A Settings Saver

A new settings saver can be created in the project window with **Right Click > Create > CitrioN > Settings Menu Creator > Settings Saver**

Available Settings Savers

Player Prefs

Saves and loads setting values from player prefs. It does not use different player prefs entries for this but instead uses a single entry which is serialized as Xml internally.



The screenshot shows a configuration window for a 'Script' named 'SettingsSaver_PlayerPrefs'. It has three fields: 'Append Data' with a checked checkbox, 'Player Prefs Key' with the text 'Settings', and a close button in the top right corner.

Append Data

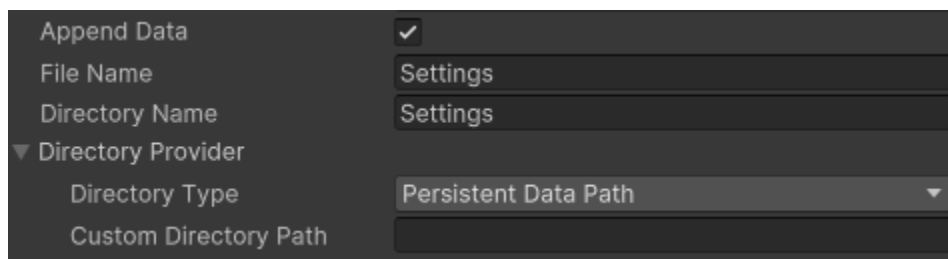
Should the existing save entry be used to add values or should a new one override the existing?

Player Prefs Key

The PlayerPrefs key to save the settings values to.

Json & Xml

Saves and loads setting values from a Json or Xml file respectively.



The screenshot shows a configuration window for a 'Script' named 'SettingsSaver_JsonXml'. It has several fields: 'Append Data' with a checked checkbox, 'File Name' with the text 'Settings', 'Directory Name' with the text 'Settings', a collapsed 'Directory Provider' section, 'Directory Type' with a dropdown menu showing 'Persistent Data Path', and 'Custom Directory Path' with an empty text field.

Append Data

Should the existing save entry be used to add values to or should a new one override the existing?

File Name

The name of your settings save file.

Directory Name

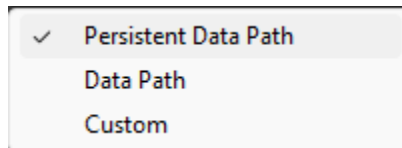
The name of the directory/folder in which the save file is stored.

Directory Provider

The provider used to determine the root save directory

Directory Type

The method used to provide the root save directory.



Persistent Data Path

Like the name suggests it uses a persistent directory path. This option is the default one and recommended as it will never change on the same device. ([Official Documentation](#)).

Data Path

Uses the path to the game data folder on the target device. ([Official documentation](#))

Custom

Uses the path specified in the custom directory path.

Custom Directory Path

A custom path to the save directory.

Writing a custom settings saver

A custom settings saver needs to derive from the `SettingsSaver` class and implement the 'SaveSettings' and the 'LoadSettings' methods. Take a look at the existing setting savers implementations for reference.

```

using CitrioN.SettingsMenuCreator;
using System.Collections.Generic;

Unity Script | 0 references
public class MySettingsSaver : SettingsSaver
{
    2 references
    public override Dictionary<string, object> LoadSettings()
    {
        // Load the setting values and returns them in a dictionary
        // where the key is the setting identifier and the value the
        // setting value.
        return new Dictionary<string, object>();
    }

    2 references
    public override void SaveSettings(SettingsCollection collection)
    {
        // Save the settings on the collection
    }

    2 references
    public override void DeleteSave()
    {
        // Delete save implementation
    }
}

```

LoadSettings

This should load the values for the settings and return them in a dictionary as shown in the image above.

SaveSettings

This should save/serialize the values of the provided SettingsCollection.

DeleteSave

This should delete the current save for the settings.

AppendData

If the setting values should be appended to any existing data. This is particularly useful if you have multiple menus accessing the same save file. How this is implemented is up to you. Take a look at the implementations of the provided setting savers for reference.