INIParser Documentation

by Project Boon

Ini files

This library handles ini files. Note that the file can be in any extension (e.g. txt) as long as the file content is format correctly.

```
[Player]
name=Arnold
avatar=2

; This section stored hi-scores
[Hi-score]
Top1=32900
Top2=12000
Top3=4700
```

How to use?

1. Add “INIParser.cs” to Unity.
2. Declare an INIParser object and use it.

```
INIParser ini = new INIParser();
ini.Open(“C:/Test.ini”);
ini.WriteValue(“Player”,“Name”,“Arnold”);
ini.Close();
```

Multiple ini files

Please note that for each INIParser instance, you can only have one ini file open at any one time, and you have to use Close( ) before you can open the next ini file.

```
INIParser ini = new INIParser();
ini.Open(“C:/Test.ini”);
ini.WriteValue(“Player”,“Name”,“Arnold”);
ini.WriteValue(“Hi-score”,“Top3”,1000);
ini.Close();

ini.Open(“C:/Test2.ini”);
ini.WriteValue(“Position”,“x”,2);
ini.WriteValue(“Position”,“y”,3);
ini.Close();
```

Methods

Open(string path)

Open an ini_file for reading and writing. If the file does not exist, a file will be created. Remember to call Close( ) once you're finished reading/writing from the .ini file to save all the changes you have made to the file.

Open(TextAsset asset)

Open a TextAsset as an ini_file. If you make any changes, a copy will be saved at Persistent Data Path. This function will always look at Persistent Data Path first to see if there's any modified copy of TextAsset before actually look at the TextAsset in the game bundle.

OpenFromString(string str)
Create an ini file from a string and open it for reading/writing. The string should be correctly formatted as an ini file (ie: with sections, line breaks keys and values) otherwise the ini file will not be created correctly. Note that this ini file is temporary, only exists in memory. However you can use ToString() to return the string of full ini file which can then be saved to a server or to disk.

string ToString(string str)
Return the string of full ini file

Close()
This method should be called the moment you are finished reading or writing to any open ini file. The ini file data is stored in the memory until this method is call, which force the data to be written to disk.

string ReadValue(string section, string key, string default)
(overload: bool, int, long, double, byte[], DateTime)
Read a value from ini_file. Should the value does not exist, (default) will be return.

WriteValue(string section, string key, string value)
(overload: bool, int, long, double, byte[], DateTime)
Write a value to ini_file

SectionDelete(string section)
Delete a whole section of an ini file, which will also delete all key-value pairs that are associated with it.

bool IsSectionExists(string section)
Check to see if a section exists in the ini file. You don't need to check to prevent errors, because if you *ReadValue* from a non-existent section, *ReadValue* will just return the default value. However, sometimes it can be useful to see if an ini file has saved specific data.

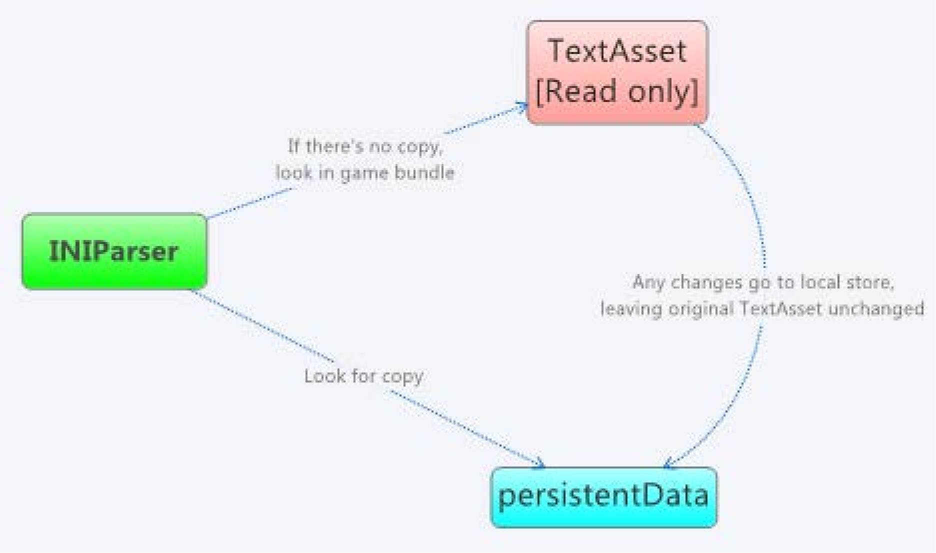
KeyDelete(string section, string key)
Delete the selected key (and its corresponding value) from the ini file.

bool IsKeyExists(string section, string key)

Check to see if a key exists in the ini file. You don't need to check to prevent errors, because if you *ReadValue* from a non-existent section, *ReadValue* will just return the default value. However, sometimes it can be useful to see if an ini file has saved specific data.

Open(TextAsset asset)

TextAsset is read-only, so that any modification is put in the sandbox area (persistentDataPath).



Example code:

```
INIParser ini = new INIParser();
TextAsset asset = Resource.Load("TextAssetExample") as TextAsset;
ini.Open(asset);
ini.WriteValue("Player","Name","Arnold");
ini.Close();
```

Sometimes, you will want to use *TextAsset* as ini file. *TextAsset* is included in the game bundle, therefore it's reliable for read/write operation on every platform. If you use streaming assets as ini files, sometimes you will be meet with read/write permission error on mobile platforms.

You have to make sure the *TextAsset* exists, otherwise any read/write operation will not work correctly.

Credits

This library is adapted from STA INIFile and modeled after Game Maker Studio INI file system.