Inifile

Unity 2019.x tested.

Author: Haikun Huang

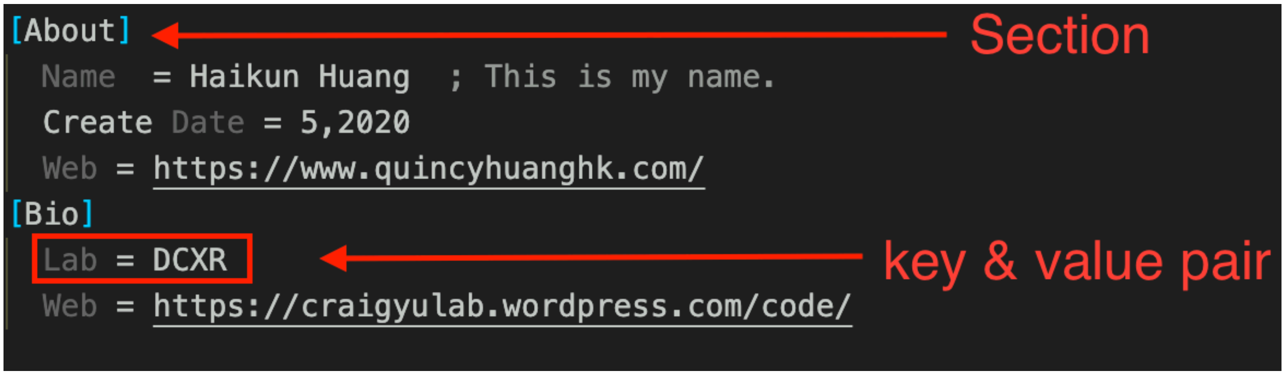
My Lab: DCXR

Github: Click Here

This tool helps to read and write a flexible and user-customized data structure from and into a plain file. This tool contains a rich set of APIs to help you to read and write the most Unity C# basic data type from and into a given plain file. You can also easily extend the basic APIs to write your own extension APIs to read and write any data type.

User-Customized Data Structure

The user-customized data structure looks like below:

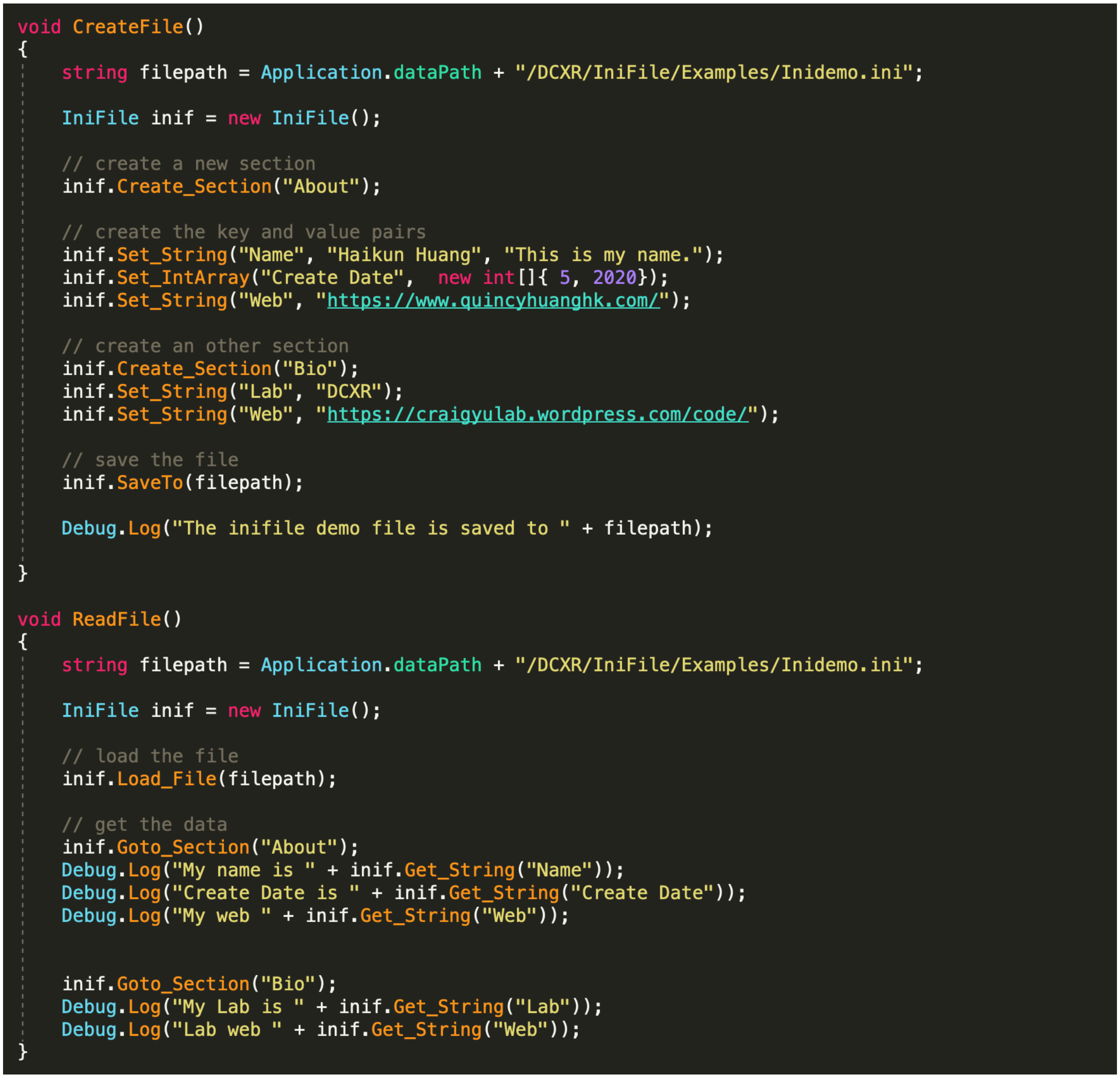


```
[About]
  Name   = Haikun Huang   ; This is my name.
  Create Date = 5,2020
  Web = https://www.quincyhuanghk.com/
[Bio]
  Lab = DCXR
  Web = https://craigyulab.wordpress.com/code/
```

It contains 2 parts, the **Sections** and the **key & value pairs (k&v pairs)**.

The **Section** is a collection of **k&v pairs**, the **k&v pair** is used to record the value by a unique key in each **Section**. And the comment of the **k&v pair** is the starting by a comma “;”.

Quick Start



```
void CreateFile()
{
    string filepath = Application.dataPath + "/DCXR/IniFile/Examples/Inidemo.ini";

    IniFile inif = new IniFile();

    // create a new section
    inif.Create_Section("About");

    // create the key and value pairs
    inif.Set_String("Name", "Haikun Huang", "This is my name.");
    inif.Set_IntArray("Create Date",  new int[]{ 5, 2020});
    inif.Set_String("Web", "https://www.quincyhuanghk.com/");

    // create an other section
    inif.Create_Section("Bio");
    inif.Set_String("Lab", "DCXR");
    inif.Set_String("Web", "https://craigyulab.wordpress.com/code/");

    // save the file
    inif.SaveTo(filepath);

    Debug.Log("The inifile demo file is saved to " + filepath);

}

void ReadFile()
{
    string filepath = Application.dataPath + "/DCXR/IniFile/Examples/Inidemo.ini";

    IniFile inif = new IniFile();

    // load the file
    inif.Load_File(filepath);

    // get the data
    inif.Goto_Section("About");
    Debug.Log("My name is " + inif.Get_String("Name"));
    Debug.Log("Create Date is " + inif.Get_String("Create Date"));
    Debug.Log("My web " + inif.Get_String("Web"));


    inif.Goto_Section("Bio");
    Debug.Log("My Lab is " + inif.Get_String("Lab"));
    Debug.Log("Lab web " + inif.Get_String("Web"));
}
```

This figure shows the most important APIs of using the **Inifile** class. More details please see the examples inside the tool.

How does it work?

The **Inifile** class works like a state machine. It has an internal curser always pointing to the current **Section**, which means, all the actions are under the current **Section**. The API **Goto_Section()** helps you to switch the **Sections**.

APIs

Basic Function

- Load a file.

  > public void Load_File(string fileName);

- Save a file.

  > public void Save();

- Save as...

  > public void SaveTo(string fileName);

- Create a new section and go to that section if it created success.

  > public bool Create_Section(string section, string comment = “”);

- Go to the section if it existed.

  > public bool Goto_Section(string section);

- Set a new string value or update its value by the given key name.

  This is the basic function as everything in plain file is the string. All the advance functions are extended from this basic function. This allows you to write your own functions by extended this function.

  > public void Set_String(string name, string value, string comment = “”);

- Get the string value by the give key name.

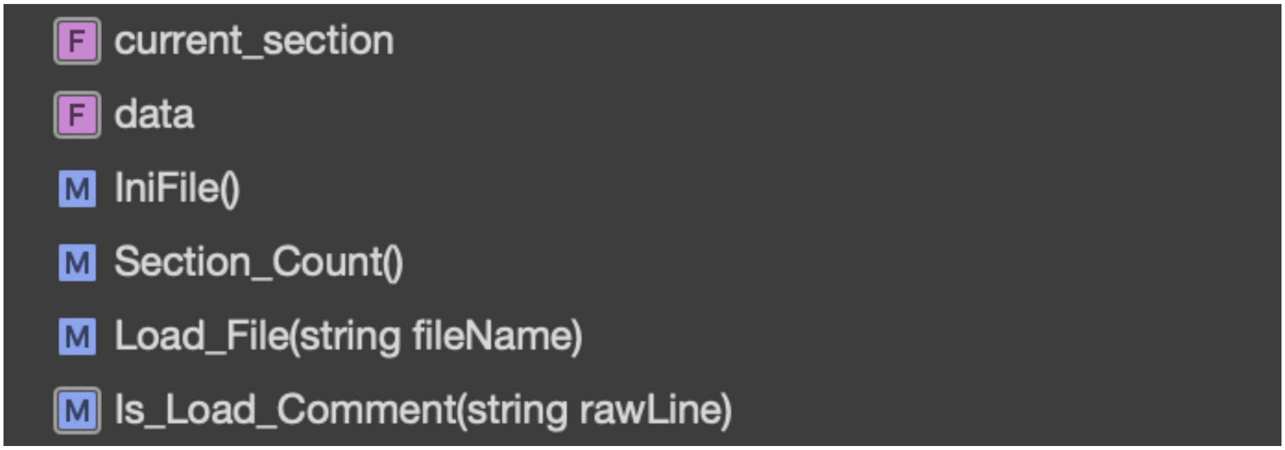
  This is the basic function as everything in plain file is the string. All the advance functions are extended from this basic function. This allows you to write your own functions by extended this function.

  > public string Get_String(string name, string defaultValue = “”);

- A set of helper functions helps you to manage your data.

  > public string[] Get_All_Section();
  >
  > public string[] Get_All_Pair_Names();
  >
  > public string[] Get_All_Pair_Values()

More functions are list below:



F current_section

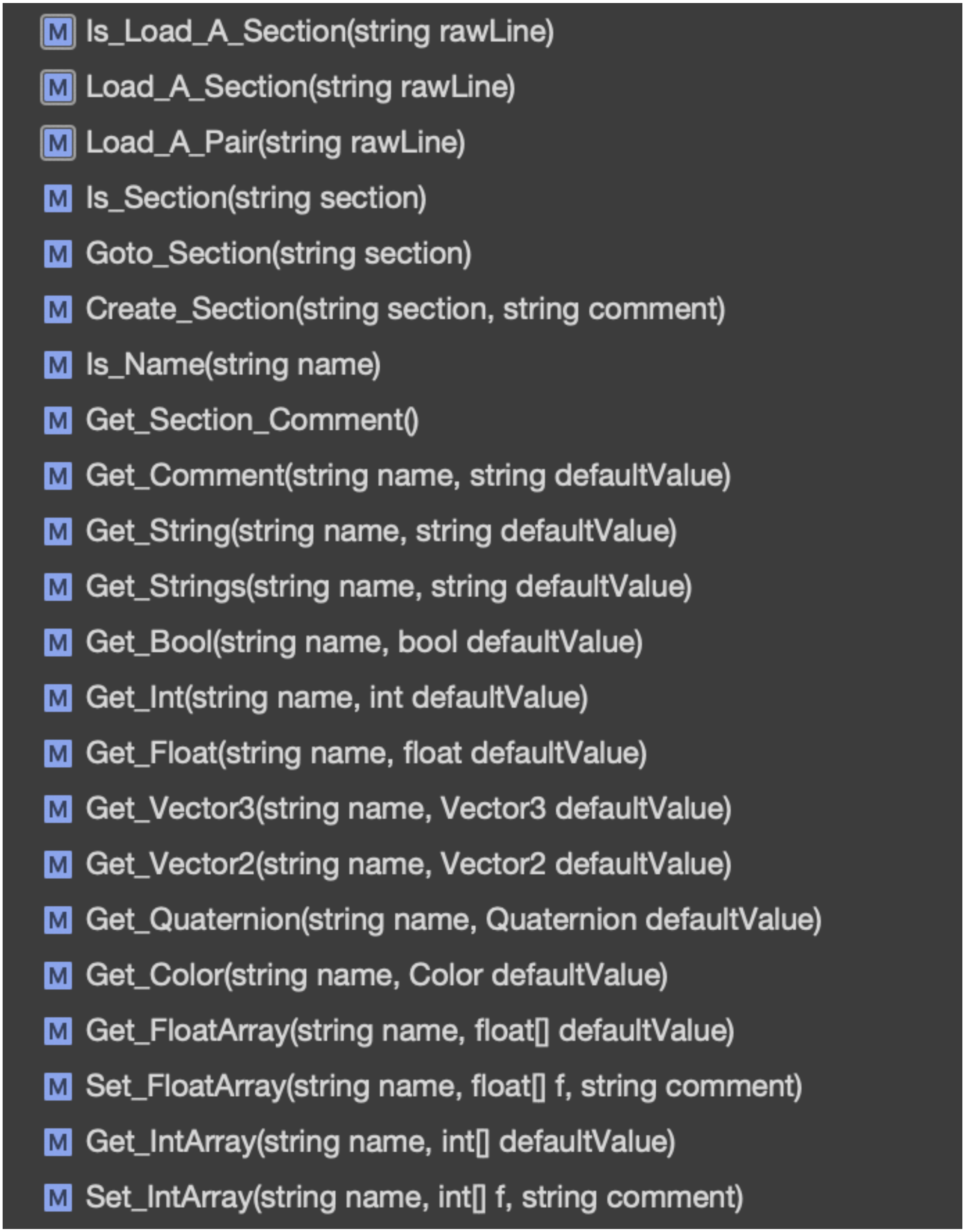
F data

M IniFile()

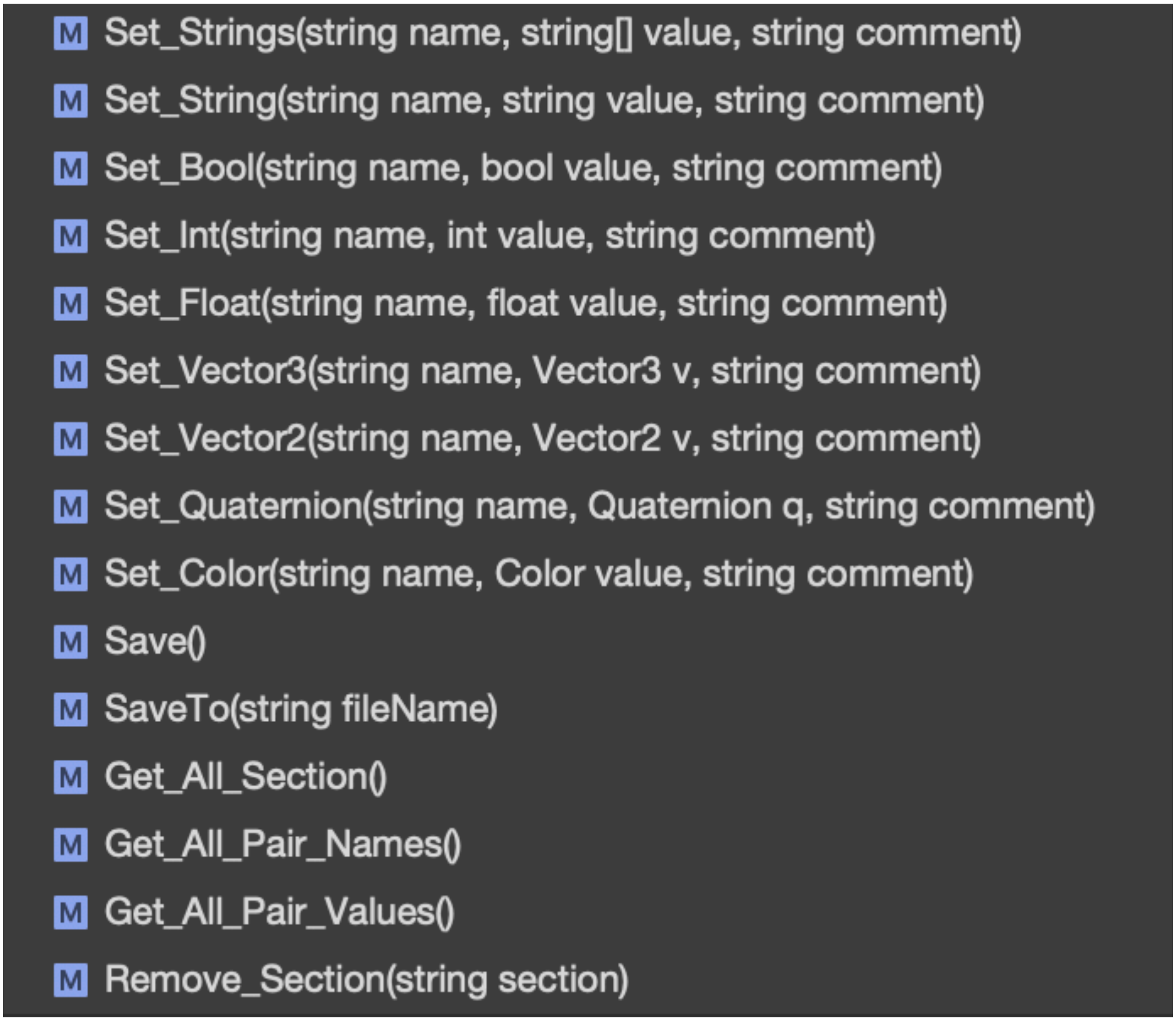
M Section_Count()

M Load_File(string fileName)

M Is_Load_Comment(string rawLine)



```
M Is_Load_A_Section(string rawLine)
M Load_A_Section(string rawLine)
M Load_A_Pair(string rawLine)
M Is_Section(string section)
M Goto_Section(string section)
M Create_Section(string section, string comment)
M Is_Name(string name)
M Get_Section_Comment()
M Get_Comment(string name, string defaultValue)
M Get_String(string name, string defaultValue)
M Get_Strings(string name, string defaultValue)
M Get_Bool(string name, bool defaultValue)
M Get_Int(string name, int defaultValue)
M Get_Float(string name, float defaultValue)
M Get_Vector3(string name, Vector3 defaultValue)
M Get_Vector2(string name, Vector2 defaultValue)
M Get_Quaternion(string name, Quaternion defaultValue)
M Get_Color(string name, Color defaultValue)
M Get_FloatArray(string name, float[] defaultValue)
M Set_FloatArray(string name, float[] f, string comment)
M Get_IntArray(string name, int[] defaultValue)
M Set_IntArray(string name, int[] f, string comment)
```



```
M Set_Strings(string name, string[] value, string comment)
M Set_String(string name, string value, string comment)
M Set_Bool(string name, bool value, string comment)
M Set_Int(string name, int value, string comment)
M Set_Float(string name, float value, string comment)
M Set_Vector3(string name, Vector3 v, string comment)
M Set_Vector2(string name, Vector2 v, string comment)
M Set_Quaternion(string name, Quaternion q, string comment)
M Set_Color(string name, Color value, string comment)
M Save()
M SaveTo(string fileName)
M Get_All_Section()
M Get_All_Pair_Names()
M Get_All_Pair_Values()
M Remove_Section(string section)
```

Advantages

- Enable you to read and write the loosen data structure.
- Good for record and read the metadata.