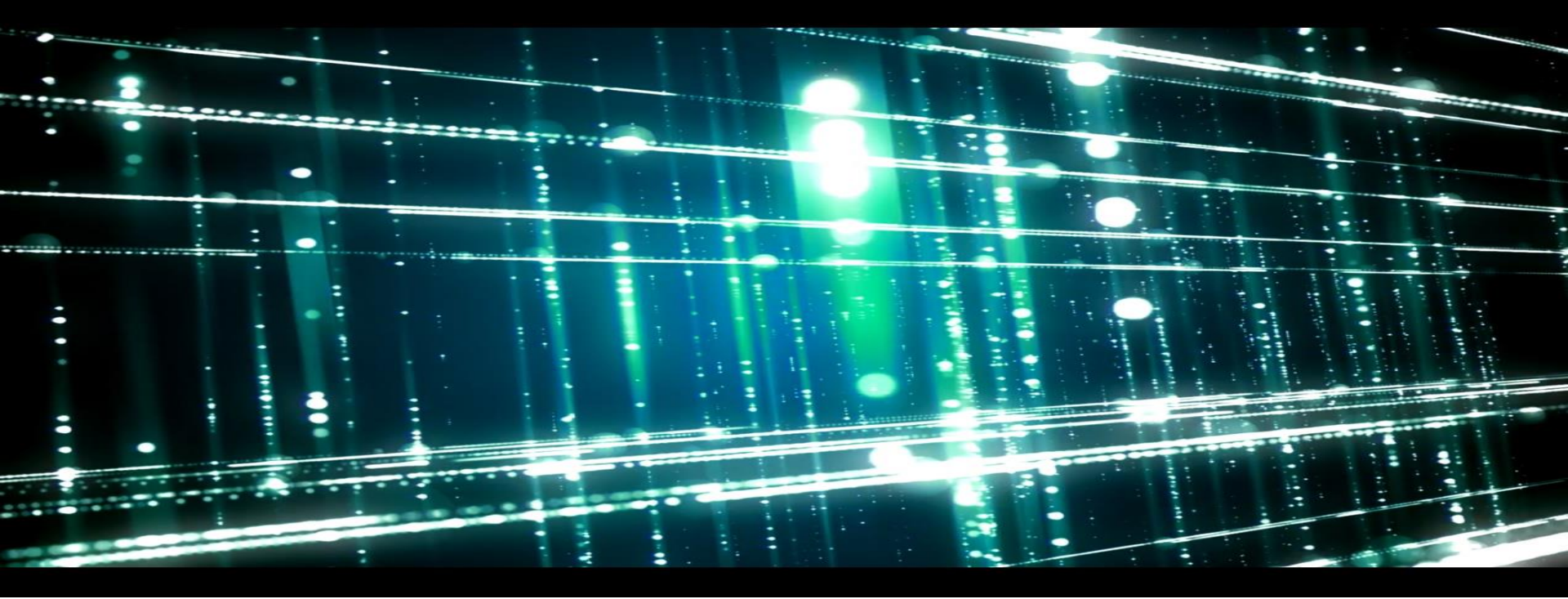


日志调试与配置管理

UI界面框架设计　讲师:刘国柱

配置管理

- 所谓“配置管理”是指一个游戏项目（软件项目），很多需要经常变化的需求或者数据，最好以配置文件的形式存在，从而代替“硬编码”方式。

例如：　游戏项目语言的国际化、
　　日志文件的保存路径等。

```
{
"ConfigInfo":
[
{"Key":"本配置文件说明,LogState 表示日志状态,

{"Key":"LogDriveName",
"Value":"F"},

{"Key":"LogPath",
"Value":"SUIFW_Log_New"},

{"Key":"LogState",
"Value":"Develop"},

{"Key":"LogMaxCapacity",
"Value":"300"},

{"Key":"LogBufferNumber",
"Value":"1"}
]
}
```

配置管理

- 目前(2017)国际国内普遍采用的配置管理方式主要有两种： XML与Json 方式。

两者各有优缺点：

XML：对于数据的精确表示、易读性很高。

微软很多的项目都内置对XML作为配置文件的支持。

（例如： 网站项目：ASP.Net、 WinForm 等）

缺点是读写速度慢，这个问题在移动端尤其突出。

Json：读写速度快，但是易读性没有XML好，但是可以接受。

所以本框架项目都采用Json作为配置文件。

Json概述

- JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式。 JSON采用完全独立于语言的文本格式，但是也使用了类似于C语言家族的习惯（包括C、C++、C#、Java、JavaScript、Perl、Python等）。这些特性使JSON成为理想的数据交换语言。 易于人阅读和编写，同时也易于机器解析和生成(一般用于提升网络传输速率)。

Json语法

- JSON 语法 （JSON 语法是 JavaScript 对象表示语法的子集）
- 特点：

  数据在键值对中，数据由逗号分隔。

  花括号保存对象，方括号保存数组。

- JSON 数据的书写格式是：名称/值对。

  "firstName":"John"

- 具体示例

```
{
    "people":[
        {"firstName":"Brett","lastName":"McLaughlin","email":"aaaa"},
        {"firstName":"Jason","lastName":"Hunter","email":"bbbb"},
        {"firstName":"Elliotte","lastName":"Harold","email":"cccc"}
    ]
}
```

Json解析(1)

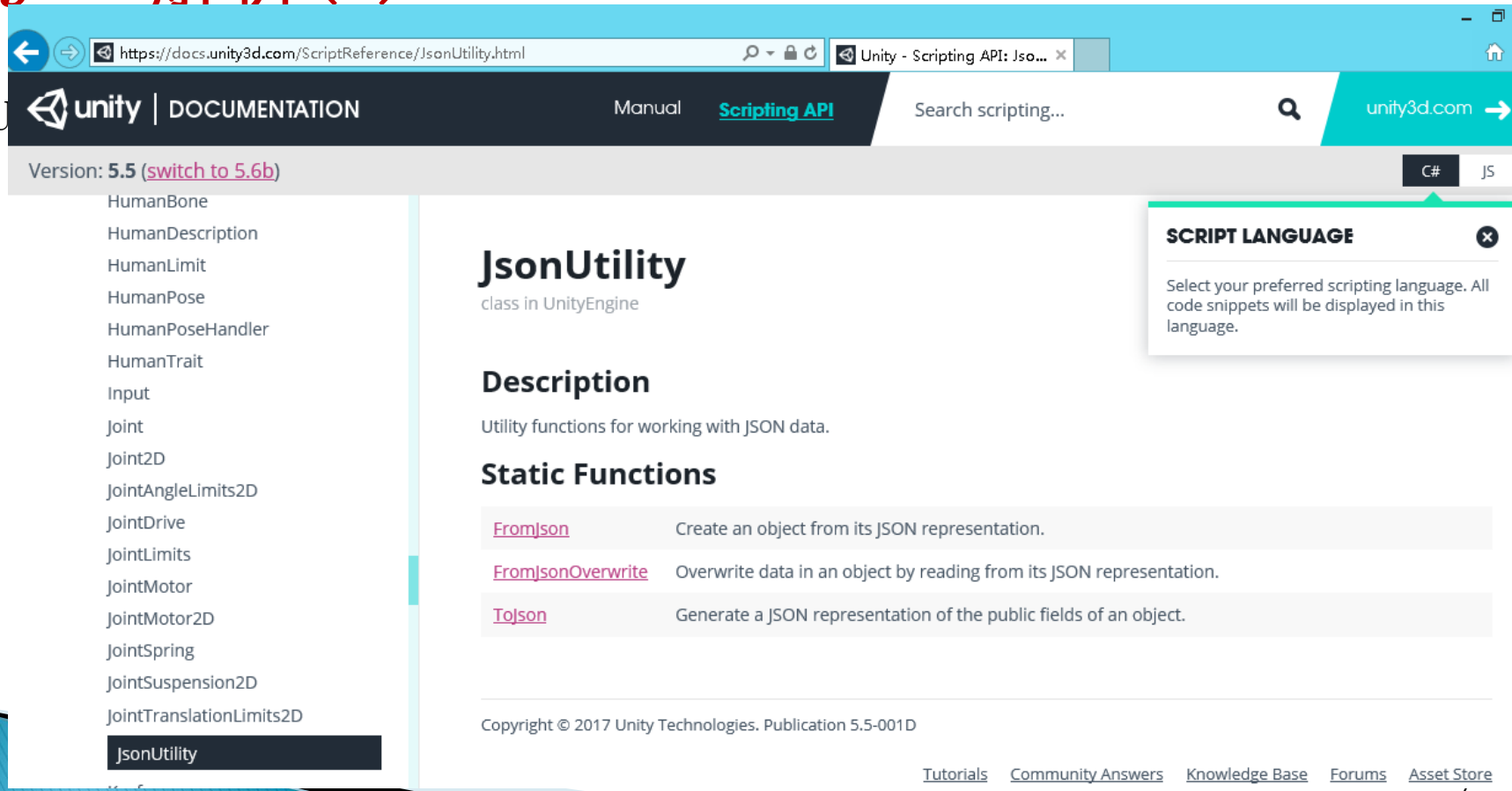
目前(2017)国际国内对于Json 的解析主要有以下几种方式

- .NET自带的运行时序列化和反序列化json工具。
  命名空间 System.Runtime.Serialization.Json
  缺点是需要编写大量代码，自己来封装一些实用方法，不推荐。

- 插件解析：
  目前国内用的最多的Json解析插件：litejson

Json解析(2)

- U


https://docs.unity3d.com/ScriptReference/JsonUtility.html
Unity - Scripting API: Jso...
unity | DOCUMENTATION
Manual
Scripting API
Search scripting...
unity3d.com
Version: 5.5 (switch to 5.6b)
C#
JS
HumanBone
HumanDescription
HumanLimit
HumanPose
HumanPoseHandler
HumanTrait
Input
Joint
Joint2D
JointAngleLimits2D
JointDrive
JointLimits
JointMotor
JointMotor2D
JointSpring
JointSuspension2D
JointTranslationLimits2D
JsonUtility
JsonUtility
class in UnityEngine
SCRIPT LANGUAGE
Select your preferred scripting language. All code snippets will be displayed in this language.
Description
Utility functions for working with JSON data.
Static Functions
FromJson
Create an object from its JSON representation.
FromJsonOverwrite
Overwrite data in an object by reading from its JSON representation.
ToJson
Generate a JSON representation of the public fields of an object.
Copyright © 2017 Unity Technologies. Publication 5.5-001D
Tutorials
Community Answers
Knowledge Base
Forums
Asset Store

Json基本解析示例

- 示例1：

  对于Unity 原生支持Json 解析方法的最简测试演示。

- 示例2：

  对于Json 文件的实战性测试用例演示。

开发Json配置管理器

- 定义通用配置管理器接口
- 开发实现IConfigManager 接口的通用配置管理器

```
{
"ConfigInfo":
[
{"Key":"本配置文件说明,LogState 表示日志状态,

{"Key":"LogDriveName",
"Value":"F"},

{"Key":"LogPath",
"Value":"SUIFW_Log_New"},

{"Key":"LogState",
"Value":"Develop"},

{"Key":"LogMaxCapacity",
"Value":"300"},

{"Key":"LogBufferNumber",
"Value":"1"}
]
}
```

```
{
"ConfigInfo":
[
{"Key":"本配置文件说明"},

{"Key":"LogonForms",
"Value":"UIPrefabs\\LogonForms"},

{"Key":"SelectHeroForms",
"Value":"UIPrefabs\\SelectHeroForms"},

{"Key":"HeroInfo",
"Value":"UIPrefabs\\HeroInfo"},

{"Key":"MainForms",
"Value":"UIPrefabs\\MainForms"},

{"Key":"MarketForms",
"Value":"UIPrefabs\\MarketForms"},

{"Key":"GoodsInfoForms",
"Value":"UIPrefabs\\GoodsInfoForms"},

{"Key":"ConfirmForms",
"Value":"UIPrefabs\\ConfirmForms"}
```

配置管理器应用

- UI管理器中关于“UI窗体预设路径”集合中，把前面“硬编码”改为应用Json 配置管理的方式。

第1步在Resources 目录下建立关于“UIFormsConfigInfo”的Json 文件。

第2步对于UIManager.cs 中的“UI窗体预设路径”集合做配置管理。

```
{
"ConfigInfo":
[
{"Key":"本配置文件说明"},

{"Key":"LogonForms",
"Value":"UIPrefabs\\LogonForms"},

{"Key":"SelectHeroForms",
"Value":"UIPrefabs\\SelectHeroForms"},

{"Key":"HeroInfo",
"Value":"UIPrefabs\\HeroInfo"},

{"Key":"MainForms",
"Value":"UIPrefabs\\MainForms"},

{"Key":"MarketForms",
"Value":"UIPrefabs\\MarketForms"},

{"Key":"GoodsInfoForms",
"Value":"UIPrefabs\\GoodsInfoForms"},

{"Key":"ConfirmForms",
"Value":"UIPrefabs\\ConfirmForms"}
```

日志调试

- 日志调试在游戏的开发全过程中占有非常重要的作用。

- “地下守护神”项目中，首次提出自定义“日志调试”脚本插件的开发思路与具体实现。

  现在为了更好的适用于PC与移动端调试的目的，进行再次重构。

  第1： 对于读写文件内部方法做重构完善，使得日志文件实现自动侦测与创建写入操作等。

  第2： 改以前的针对XML的配置文件的读取方式为Json 文件的读取。

  目的一： 提高读取速度。

  二： 更好的应用在移动端的部署


谢谢大家！
刘老师讲Unity系列教程