火车浏览器 V7 二次开发 SDK（C#）

一、开发前准备工作

1. 解压示例项目 DemoSDK

示例项目演示了所有的调用方式和代码，直接可以上手使用。使用默认的测试帐号，脚本运行器可以显示窗口并运行 3 分钟。如果使用的是加密狗版或无限授权，请将授权 AssemblyInfo.cs 替换掉默认的该文件。

2. 复制两个文件

复制 Runtime 目录下 LpSdk.dll 和 LpCore.dll 两个文件至浏览器根目录下

3. 打开示例项目 DemoSDK

请修改 DemoSDK 项目生成地址为浏览器根目录，然后开始测试。

二、配置类库 TaskConfig

```
namespace LpSdk
{
    /// <summary>
    /// 任务配置文件，命令行或是直接类库操作需要传入的
    /// </summary>
    public class TaskConfig
    {
        /// <summary>
        /// 任务运行名称
        /// </summary>
        public string TaskName = "";

        /// <summary>
        /// 任务脚本文件，不得为空
        /// </summary>
        public string TaskFile = "";

        /// <summary>
        /// Skey 参数
        /// </summary>
        public string SKey = "";

        /// <summary>
        /// 浏览器缓存文件目录,不指定系统自动创建
        /// </summary>
```

```
public string CacheDir = "";

/// <summary>
/// 浏览器窗口显示还是隐藏
/// </summary>
public bool IsShowWindow = true;

/// <summary>
/// 是否禁音
/// </summary>
public bool IsMute = true;

/// <summary>
/// 设置变量的值
/// </summary>
public List<VariableInfo> VarInfo = new List<VariableInfo>();

/// <summary>
/// 日志保存文件地址
/// </summary>
public string LogFilePath = "";

/// <summary>
/// 结果文件保存地址
/// </summary>
public string ResultFile = "";

/// <summary>
/// 代理 ip
/// </summary>
public string ProxyIP = "";

/// <summary>
/// 代理端口
/// </summary>
public int ProxyPort = 0;

/// <summary>
/// 代理用户名
/// </summary>
public string ProxyUserName = "";

/// <summary>
/// 代理密码
```

```
        /// </summary>
        public string ProxyPassword = "";
    /// <summary>
        /// 将配置文件生成 xml 并 urlencode 处理
        /// </summary>
        /// <returns></returns>
        public override string ToString()
        {
             return something;
        }

        public string ToXmlString()
        {
            return Lv.Crypt.Base64_Decode(this.ToString());
        }

        public static TaskConfig FromBase64String(string base64)
        {
            string xml = Lv.Crypt.Base64_Decode(base64);
            return FromXmlString(xml);
        }

        public static TaskConfig FromXmlString(string xml)
        {
            return taskConfig;
        }
    }
}
```

三、运行类库 RunTask

可以通过 SetTaskProcessAction 实时的看到任务运行的细节。可以通过 SetTaskEndAction 得到最终的结果。 其中，启动任务中的 StartTask 中的 TaskConfig,包含了启用一个任务所有的信息，其中 SKey 是 sdk 的 key 之一，请注意要写上该值。AsyncStartTask 异步执行，可以得到脚本运行器的进程 id.

```
namespace LpSdk
{
    /// <summary>
    /// 通过 C#的 SDK 运行任务
    /// </summary>
    public class RunTask
    {
        /// <summary>
```

```
        /// 获取脚本列表，是完整路径，获取的是默认 projects 目录下脚本
        /// </summary>
        public static Func<List<string>> GetScripList;

        /// <summary>
        /// 得到所有变量列表,传入的是脚本地址
        /// </summary>
        public static Func<string, List<VariableInfo>> GetScriptVarList;

        /// <summary>
        /// 日志的注册事件,参数为进程 id，步骤 id,步骤名，操作类型，日志，是否成功
        /// </summary>
        public static Action<int, int, string, LpSdk.Enum.ActionType, string, bool>
SetTaskProcessAction;

        /// <summary>
        /// 脚本任务运行完成通知进程 id,结束类型，所有变量，返回字符串
        /// </summary>
        public static Action<int, LpSdk.Enum.EndType, List<LpSdk.VariableInfo>, string>
SetTaskEndAction;

        /// <summary>
        /// 开始运行任务
        /// </summary>
        public static Action<TaskConfig> StartTask;

        /// <summary>
        /// 异步开始任务,返回进程 id
        /// </summary>
        public static Func<TaskConfig, int> AsyncStartTask;
    }
}
```