《Chrome 外部填表详解-入门篇》

1. 《Chrome DevTools Protocol》简介

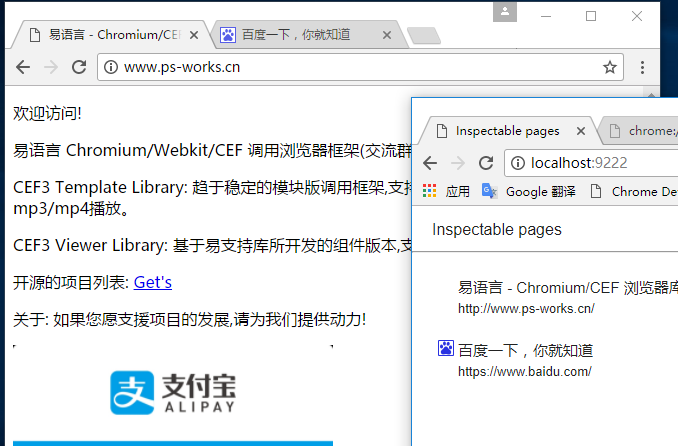
《Chrome DevTools Protocol》可以允许除《Chrome DevTools》以外的应用对Chromium或者Chrome和其它基于Blink的浏览器进行检测,检查,调试和配置Web程序,许多现有项目目前正在使用这个协议,当然也包括《Chrome DevTools》。

1. 如何开启Chrome的远程调试模式

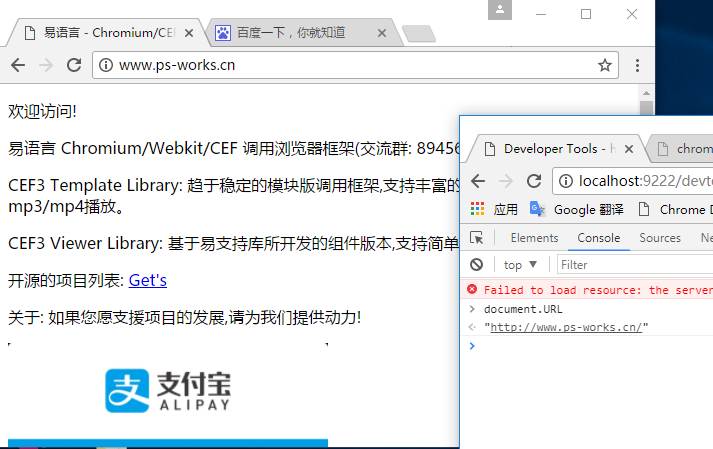
在命令行启动一个Chrome实例,并加上下面的参数,并等待其启动完成。

chrome.exe --remote-debugging-port=9222

这之后打开<http://loacalhost:9222> 就能看到你开启的Chrome实例中所有打开的标签页面



点击其中一个标签,即可开始调试目标页面



1. 导航标签(TAG)的访问与控制

获取所有打开的标签的信息,访问

http://loacalhost:9222/json

将会得到一个JSON数组,在这个数组包含了标签的一些具体信息

[ {

"description": "",

"devtoolsFrontendUrl": "/devtools/inspector.html?ws=localhost:9222/devtools/page/B7F2D344-197B-4F57-A945-939F29AE2922",

"id": "B7F2D344-197B-4F57-A945-939F29AE2922",

"title": "localhost:9222/json",

"type": "page",

"url": "http://localhost:9222/json",

"webSocketDebuggerUrl": "ws://localhost:9222/devtools/page/B7F2D344-197B-4F57-A945-939F29AE2922"

}, {

"description": "",

"devtoolsFrontendUrl": "/devtools/inspector.html?ws=localhost:9222/devtools/page/575FBA99-453B-40F1-9112-139D995246D6",

"faviconUrl": "https://www.baidu.com/favicon.ico",

"id": "575FBA99-453B-40F1-9112-139D995246D6",

"title": "百度一下，你就知道",

"type": "page",

"url": "https://www.baidu.com/",

"webSocketDebuggerUrl": "ws://localhost:9222/devtools/page/575FBA99-453B-40F1-9112-139D995246D6"

}, {

"description": "",

"devtoolsFrontendUrl": "/devtools/inspector.html?ws=localhost:9222/devtools/page/477810FF-323E-44C5-997C-89B7FAC7B158",

"id": "477810FF-323E-44C5-997C-89B7FAC7B158",

"title": "Worker pid:12089",

"type": "service\_worker",

"url": "https://www.google.com.sg/\_/chrome/newtab-serviceworker.js",

"webSocketDebuggerUrl": "ws://localhost:9222/devtools/page/477810FF-323E-44C5-997C-89B7FAC7B158"

} ]

Description: 与标签相关的描述信息

DevtoolsFrontendUrl: 与这个标签相关联的页面的前端调试地址,可将该URL复制到浏览器中直接访问即可调试目标页面

Id: 用于标识和定位这个标签的唯一身份信息

Title: 当前标签上显示的标题信息

Type: 标签的类型

Url: 与标签相关联的页面所加载的URL地址

WebSocketDebuggerUrl: 与这个标签相关联的页面的后端调试地址(第三方接口)

新建一个标签页,空白页或者带参数默认加载URL,访问

http://localhost:9222/json/new

http://localhost:9222/json/new?http://www.baidu.com

将会得到一个JSON对象,包含了新标签的一些相关信息

{

"description": "",

"devtoolsFrontendUrl": "/devtools/inspector.html?ws=localhost:9222/devtools/page/575FBA99-453B-40F1-9112-139D995246D6",

"faviconUrl": "https://www.baidu.com/favicon.ico",

"id": "575FBA99-453B-40F1-9112-139D995246D6",

"title": "百度一下，你就知道",

"type": "page",

"url": "https://www.baidu.com/",

"webSocketDebuggerUrl": "ws://localhost:9222/devtools/page/575FBA99-453B-40F1-9112-139D995246D6"

}

字段含义同上

关闭标签,需要传入该标签的ID

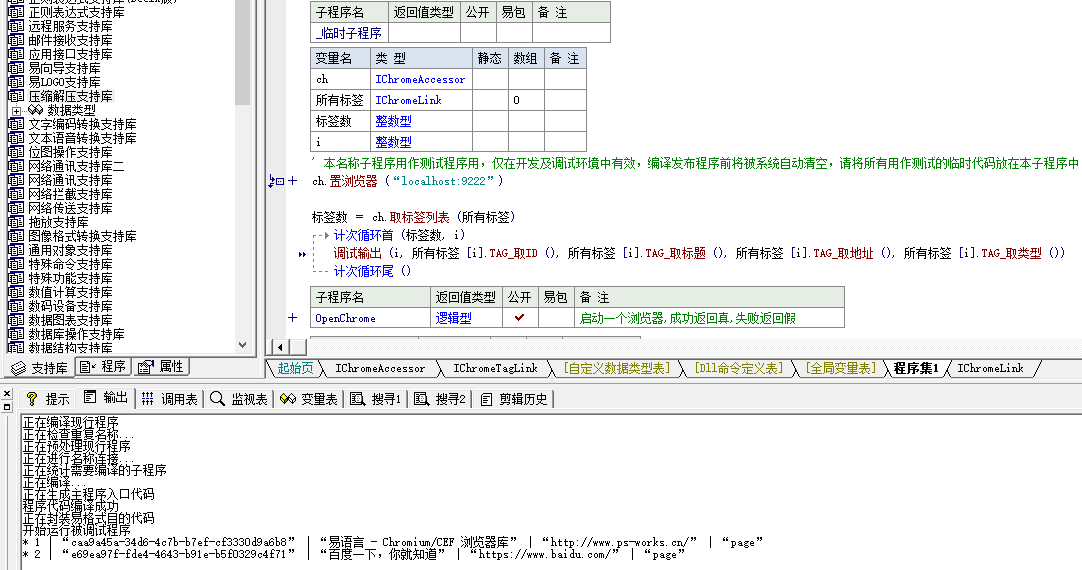
http://localhost:9222/json/close/575FBA99-453B-40F1-9112-139D995246D6

激活标签,需要传入该标签的ID

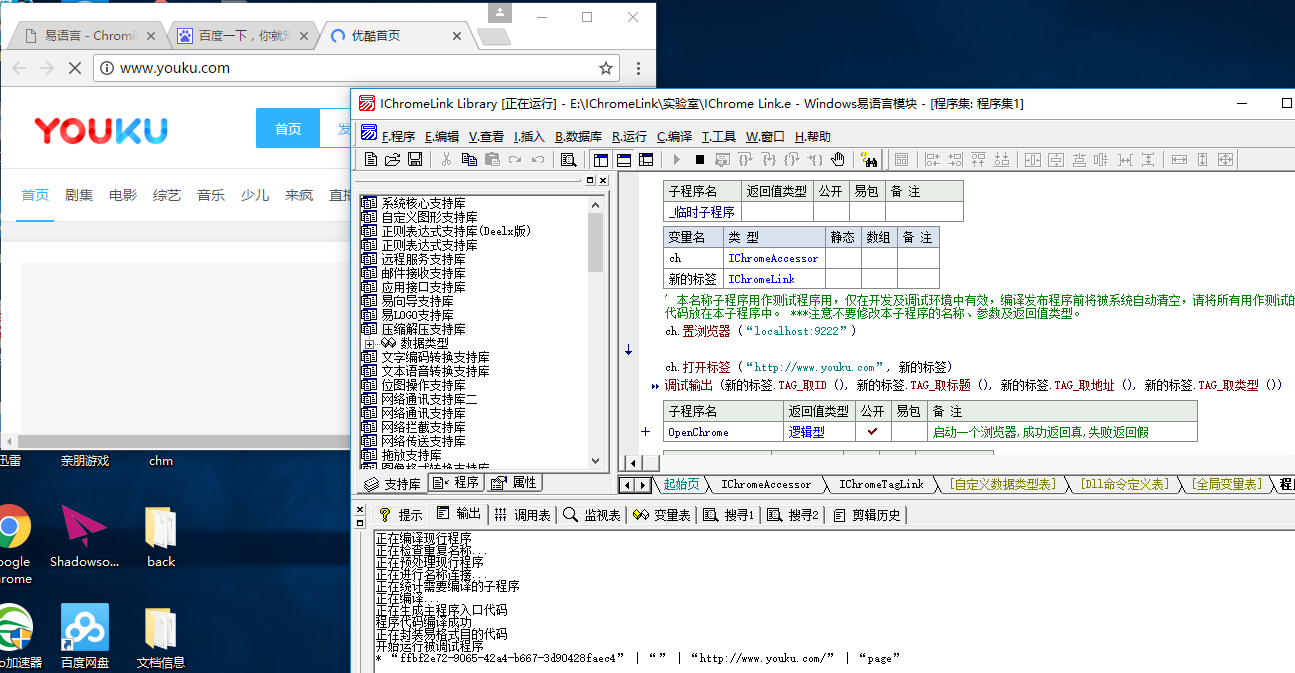
http://localhost:9222/json/activate/575FBA99-453B-40F1-9112-139D995246D6

代码方式:

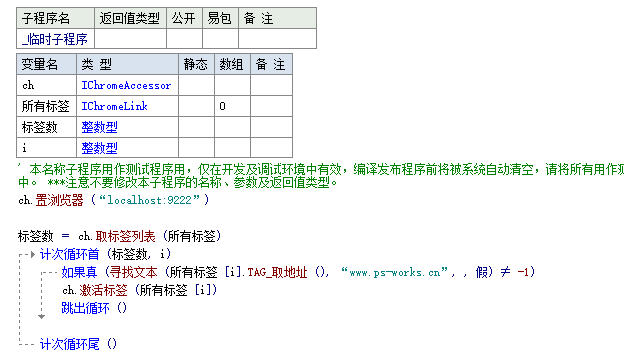
获取所有打开的标签



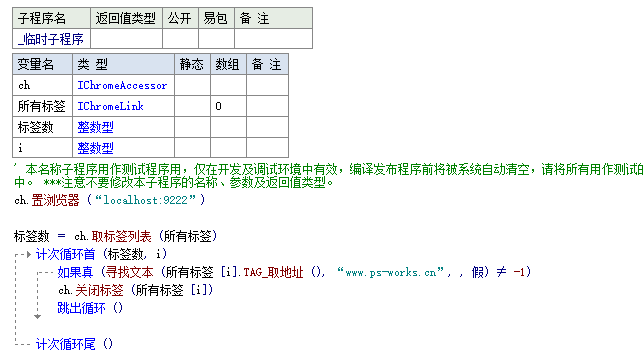
打开一个新的标签



激活指定的标签



关闭指定的标签



1. 页面的访问与控制

标签.TAG\_跳转()

//将当前页面导航至指定的URL

标签.TAG\_重新载入()

//重新加载给定页面可选地忽略高速缓存。

标签.TAG\_拍摄快照()

//捕获页面截图

标签.TAG\_清除缓存()

//清除浏览器缓存。

标签.TAG\_是否已就绪()

//如果当前页面正在加载资源中将返回真

标签.TAG\_取网页源码()

//获取当前页面中的HTML代码

标签.TAG\_执行表达式()

//在指定的页面环境中执行表达式运算(Java Script)

1. 页面中的框架结构

在Chrome中每一个页面都由一个主框架(Main Frame)和若干个子框架(frame/iframe)组成,在框架中则包含着具体的网页,要想获得合适的数据,则需匹配到对应的框架。

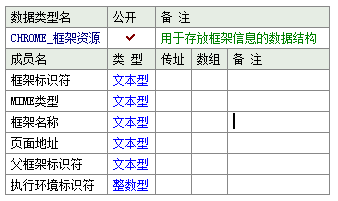
框架相关的API：

标签.TAG\_取框架数量()

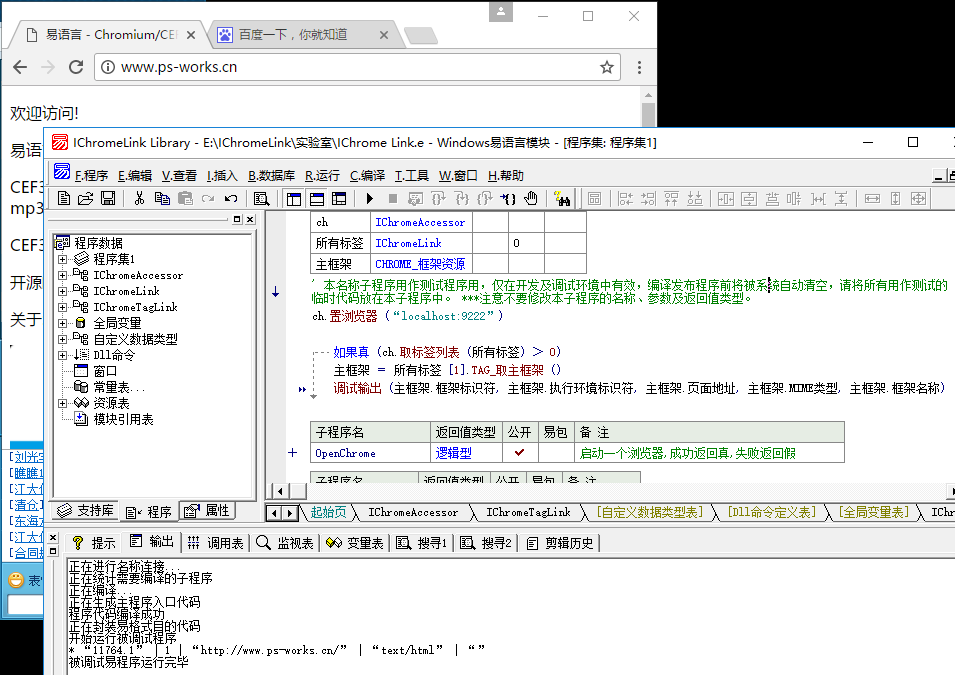
标签.TAG\_枚举框架()

标签.TAG\_取主框架()

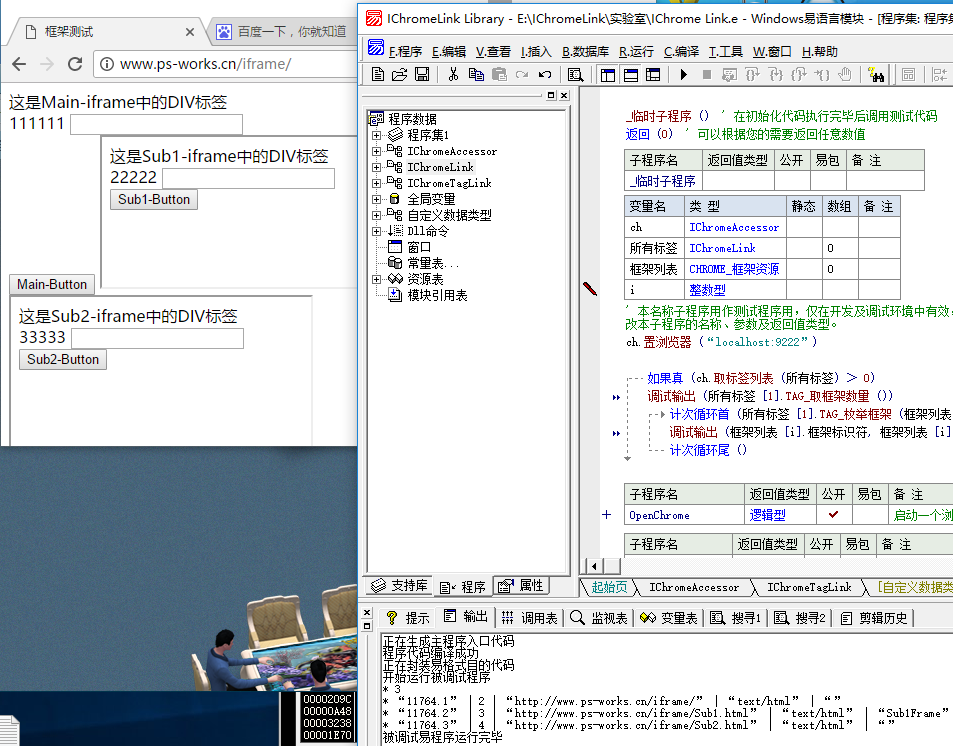
数据类型:



获取主框架:

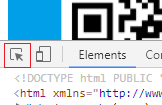


获取子框架:

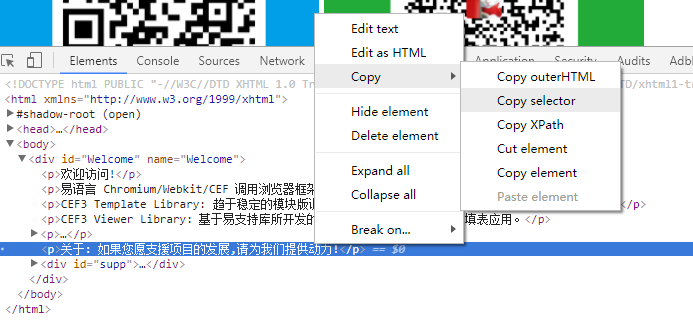


1. 了解CSS 选择器

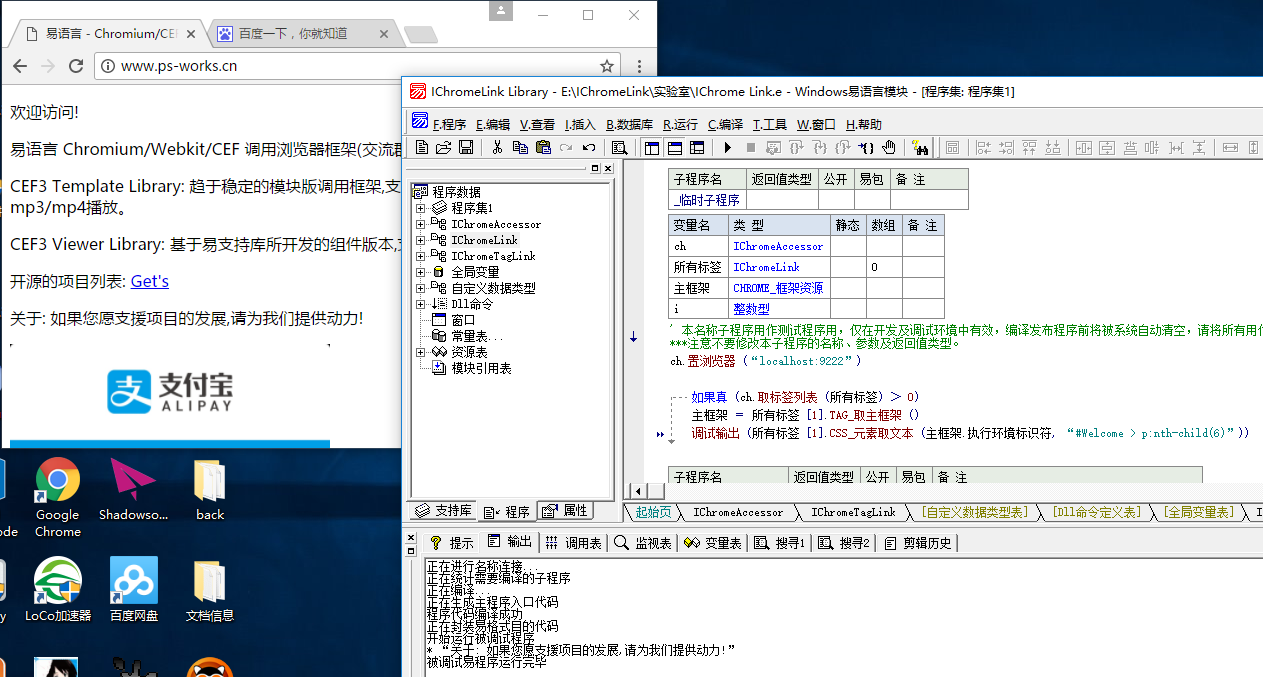
语法参考: <http://www.w3school.com.cn/cssref/css_selectors.asp>

在Chrome中,按F12打开开发者工具,点击激活后,点击页面上的任意一个元素,开发者工具将会自动定位到该元素所在源码中的位置,

在元素上右键->Copy->Copy Selector



在代码中使用:



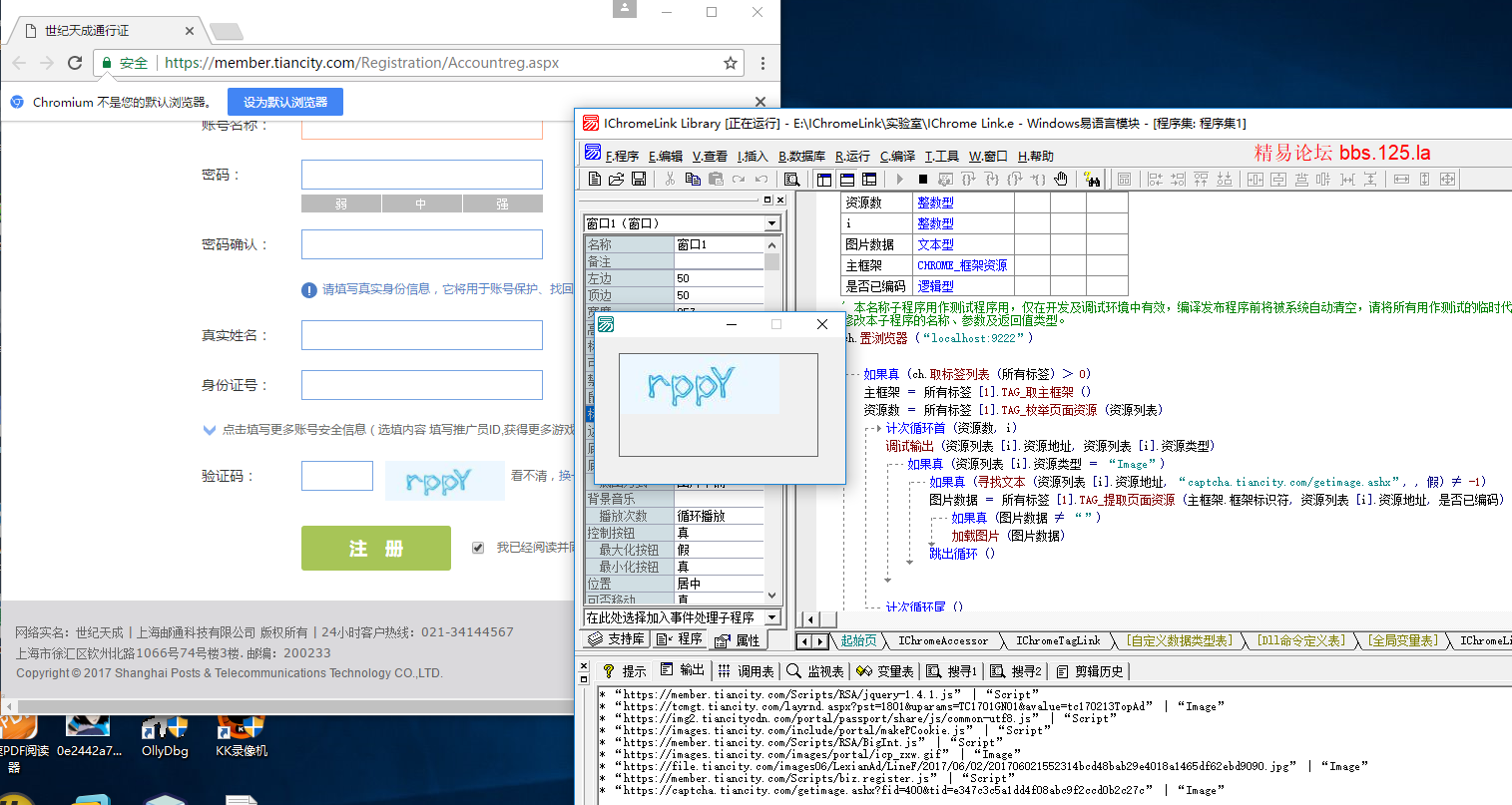
1. 如何提取网页上的任意资源(比如验证码图片)

使用到的API:

标签.TAG\_枚举页面资源()

标签.TAG\_提取页面资源()

标签.TAG\_枚举页面资源() 函数可以获得页面中所有加载的资源的信息,找到合适的资源后,在通过使用标签.TAG\_提取页面资源() 将指定的资源数据提取出来



1. 了解 Cookie相关的操作

使用到的API:

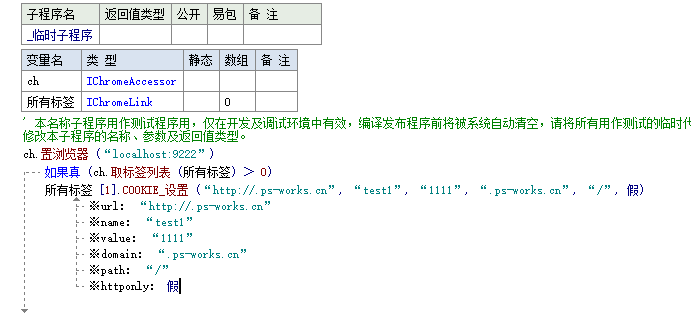
标签.COOKIE\_枚举()

标签.COOKIE\_清空()

标签.COOKIE\_删除()

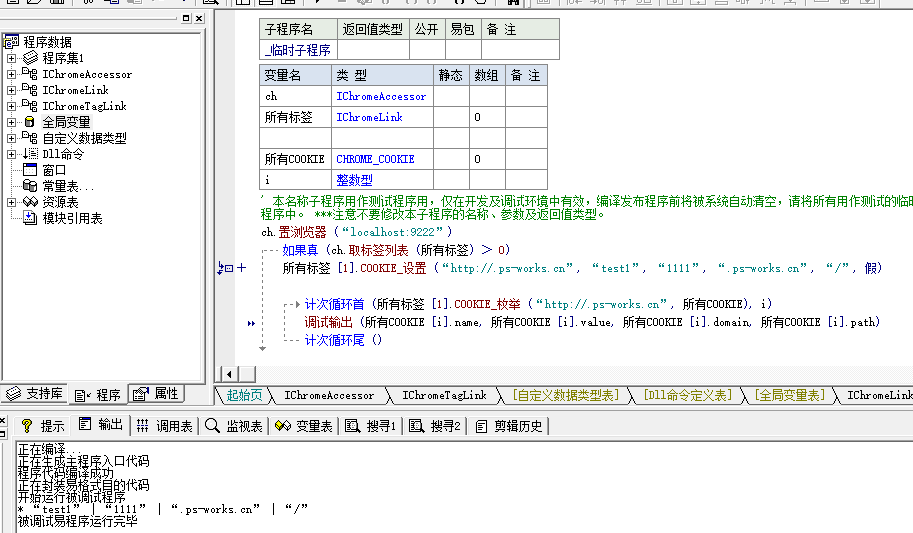
标签.COOKIE\_设置()

COOKIE的添加:

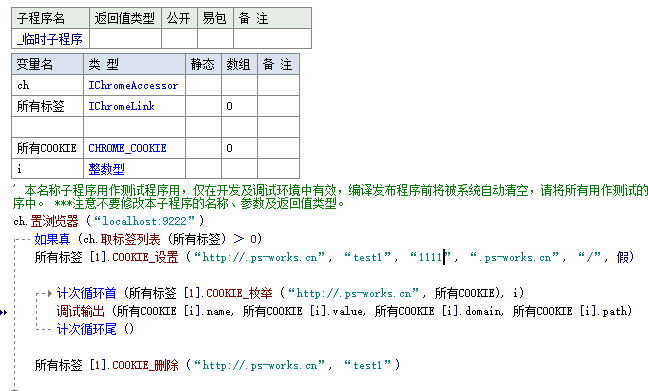


注: url参数由 http:// 或 https:// + domain 组成

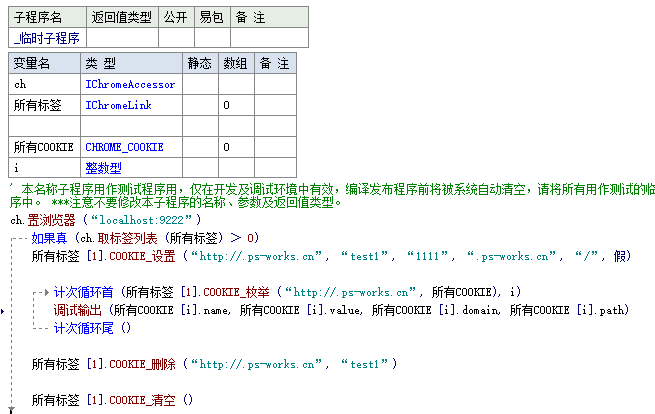
COOKIE的枚举:



COOKIE的删除:



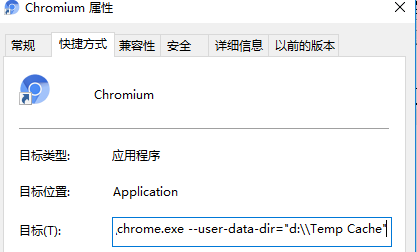
COOKIE的清空:



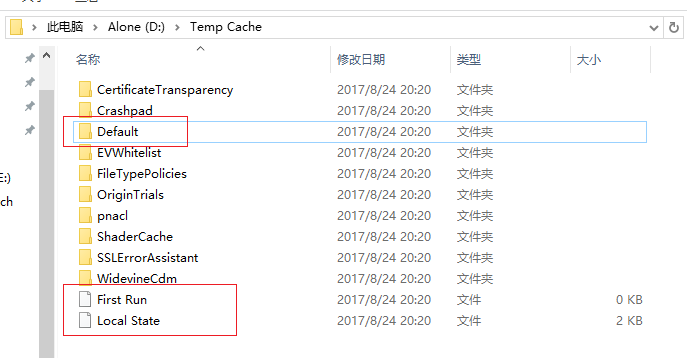
1. 如何实现浏览器的多开(独立缓存,独立COOKIE)

第一步,添加下面命令行参数,并启动Chrome 浏览器,等待其启动完毕之后,在关闭浏览器

chrome.exe --user-data-dir=”d:\\Temp Cache”



第二步,打开文件夹”d:\\Temp Cache”,复制并保存一些浏览器的初始化文件:



复制圈住文件,到另外一个文件夹备份



第三步,使用函数 OpenChrome() 启动浏览器



如果需要启动多个浏览器则修改调试端口,和缓存目录即可

1. 实例1(网站注册机)

参考附带例程: 乐都注册.e