aardio 范例: C# 使用 dynamic 调用 aardio

```
//C# 使用 dynamic 调用 aardio

/*
只要简洁，不求完美!
Win7 在市场上已经接近消失，现在开发软件再处处考虑 Win7 兼容是不必要的。
Win10 已自带.NET 4.6 以上，而 .NET 4.x 都支持 dynamic 对象
*/
import dotNet.v4;
var compiler = dotNet.createCompiler("C#"); //创建C#编译器

//必须引入下面的几个 DLL
compiler.Reference(
    "System.Core.dll","Microsoft.CSharp.dll"
);

compiler.Source = /******
using System;
using System.Dynamic;

namespace CSharpLibrary
{
    public class Object
    {
        //只要将 aardio 对象类型写为 dynamic 就可以自由调用了。
        public object Hello( dynamic aardioObject, dynamic aardioFunction ){

            //调用 aardio 函数
            var num = aardioFunction(12,3,this);

            //修改 aardio 对象属性
            aardioObject.属性名 = num;

            //调用 aardio 对象函数
            var ret = aardioObject.执行aardio(" console.log('在C#中执行aardio代码') ",this);

            //读 aardio 对象属性
            return aardioObject.test.abc;
        }
    }
}
******/

compiler.import("CSharpLibrary"); //自程序集导入名字空间

//使用 C# 编写的类构造对象实例
var cSharpObj = CSharpLibrary.Object();

var aardioObject =  {
    test = { abc = 12345;  }
    属性名 = 123;
    执行aardio = function( code,netObj ){

         /*
        aardio 函数传给 .NET 时（委托、事件），
        aardio 会在原函数外包装一层代理函数，并自动处理回调参数。
        回调 aardio 函数的参数中的 .NET 对象会自动封装为 dotNet.object 。

        但 .NET 反过来直接回调 aardio 对象的成员函数时，
        回调 aardio 函数的参数中的 .NET 对象没有经过处理，不会自动封装为 dotNet.object 。

        原生 .NET 对象在 aardio 中存为 COM 对象，
        而原生 COM 接口不支持 .NET 的很多特性，例如重载（函数名相同，参数不同）。

        .NET 原生对象传入 com.IsNetObject() 会返回非 0 值。
        而 dotNet.object 传入 dotNet.getObject() 会返回非 null 值。

        解决方案：
        如下将原生 .NET 对象转换为 aardio 中的 dotNet.object 。
        */
        netObj = dotNet.object(netObj);

        loadcode(code)();
        return 123;
    }
}
```

```
import console;
//调用实时编译的 C# 函数
var ret = cSharpObj.Hello(
    aardioObject,
    function(a,b,netObj){
        /*
        这虽然是通过 dynamic 回调的函数。
        但 aardio 函数作为参数传递给 .NET 时，aardio 会自动封装并转换所有回调参数。
        所以这里的 netObj 是 dotNet.object 对象，不需要转换。
        */
        console.log("aardio 函数被 C# 调用了,参数:",a,b)

        //无论.NET 回调是否在同一线程，被 .NET 回调的 aardio 函数总是在原调用线程执行（不必考虑多线程规则与同步）。

        return a + b;
    }
);

console.dump(ret);
console.dumpJson(aardioObject);
console.pause();
```

Markdown 格式