aardio 范例: 在 aardio 中访问 .NET 下标 Item[]

```
//在 aardio 中访问 .NET 下标 Item[]
import console;
import dotNet;
var compiler = dotNet.createCompiler("C#");

compiler.Source = /******
using System;
using System.Collections.Generic;
namespace CSharpLibrary
{
    public class TestClass
    {
        private Object [] values = new Object [] {1,2,3,4,5,6,7,8,9};
        public Object this [int index]
        {
            get  { return values[index]; }
            set  { values[index] = value; }
        }

        public Dictionary<string,string> dict = new Dictionary<string,string> ();
   }
}
******/

compiler.import("CSharpLibrary"); //编译并引入 C# 名字空间
var netObj = CSharpLibrary.TestClass(); //使用 C# 编写的类构造对象实例

/*
如果使用索引下标操作符 [] 获取成员,
则 aardio 与 C# 一样解析为读取 .NET 对象的 Item[] 属性。
如果 a["b"] 读取值失败且对象 .NET 对象不存在 Item[] 属性,则改为执行 a.b 读取属性。
*/
netObj.dict["test"] = "abc";//写字典的键值
console.log( netObj.dict["test"] );//读字典的键值

//直接读写 Item 属性也可以。
var item = netObj.Item[5];//注意这里要按 C# 起始下标为 0 （ aardio 对象 起始下标为 1）。
var item = netObj.getItem(5); //这样读下标属性也可以,支持多参数。
var item = netObj.Item(5); //get 前缀可以省略,支持多参数。
netObj.Item[5] = 123;
netObj.setItem(5,123);

//如果省略属性名 Item,则起始下标为 1。
var item = netObj[6];//等价于 netObj.Item[6-1]。
netObj[6] = 123;

/*
极罕见的 C# 对象也有可能自定义起始下标为 1,
如果写 netObj[2] 会很奇怪,可以改为 netObj.Item[1] 。
*/
console.pause();
```

Markdown 格式