

[首页](#) [新随笔](#) [联系](#) [订阅](#) [管理](#)

## 公告

[计划  
个月](#)

## 最新随笔

[ounce和throttle](#)[pt实现简单的双向绑定](#)[图了解Redux中的重要概念](#)[-Flask搭建一个记录工具](#)[vaScript模式](#)[的Python陷阱和注意点](#)[Script继承的那些事](#)[pt中的函数表达式](#)[JavaScript原型](#)[TCP: 总结和索引](#)

## 随笔分类

[pth笔记\(15\)](#)[\(3\)](#)[5\)](#)[by Step\(8\)](#)[SS](#)[on\(1\)](#)[pt\(10\)](#)[管理和多环境\(2\)](#)[3\)](#)[通信\(3\)](#)[\(2\)](#)

## IronPython和C#交互

IronPython是一个.NET平台上的Python实现引擎与运行时支持，能够与.NET已有的

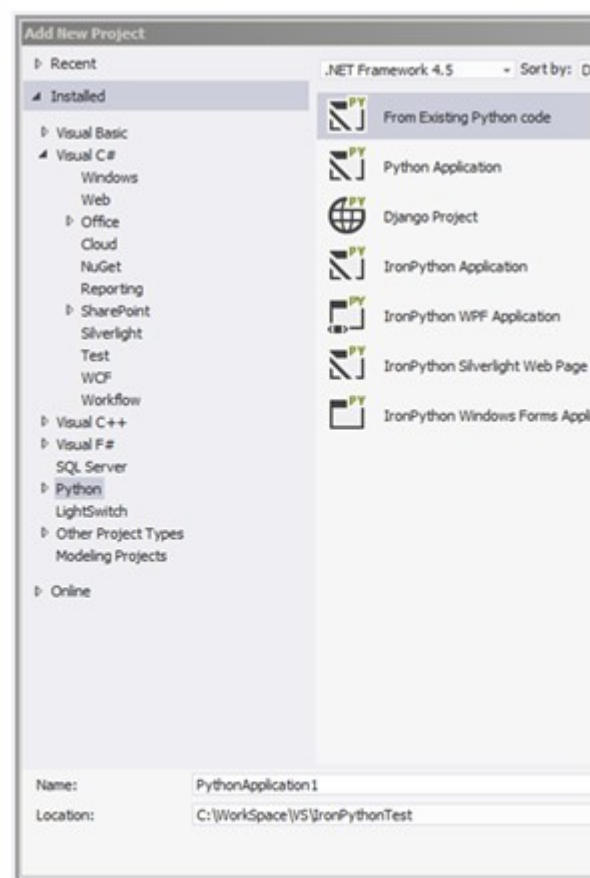
IronPython已经很好的集成到了.NET framework C#的交互也就变得很简单了。下面就通过IronPython和C#之间的交互。

### 环境设置

工欲善其事，必先利其器，所以在开始IronPython先找到一个方便的开发环境。

PTVS ( Python tools for Visual Studio ) 是一个插件，支持 VisualStudio 2010/2012/2013，安装后就可以直接通过VS进行IronPython的开发了。

下面一个截图显示了我们可以新建的项目：



### IronPython调用C#

[on笔记\(16\)](#)[\(1\)](#)[godb\(12\)](#)

## 最新评论

[MongoDB索引](#)

MongoDB的书，单个索引一定选择选择性立索引，在复合索引的时候发现一个问题：对选择性低的字段建立索引，在对选择性立索引，为什么我这么说呢？比如学号和合.....

--AK47Sonic

[Step by Step - \(1\) Git 简介](#)

--无情的雨和无情的你

[Python深拷贝和浅拷贝](#)

--feng\_yy

[Python装饰器](#)

题，博主的文章通过微信分享到手机只能容竖屏。

--天外归云

[Python装饰器](#)

，看过的写的关于装饰器最好的一篇。

--天外归云

## 阅读排行榜

[格式化字符串\(35975\)](#)[JavaScript原型\(10020\)](#)[包管理工具\(8225\)](#)[Python深拷贝和浅拷贝\(8105\)](#)[编译器和生成器\(4873\)](#)

## 评论排行榜

[JavaScript原型\(32\)](#)[Step by Step - \(1\) Git 简介\(22\)](#)[Python中的this\(9\)](#)[Python的执行上下文\(9\)](#)

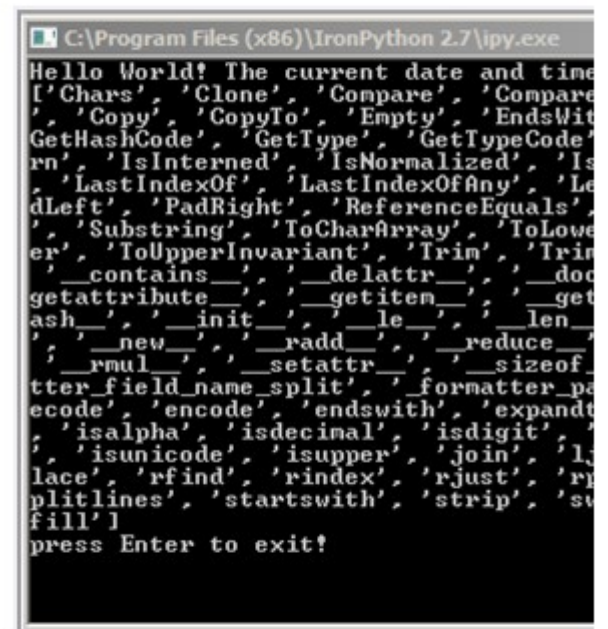
首先我们看下如何在IronPython中使用C#的

### 使用标准.NET库

在.NET中，有很多标准库，在IronPython中这些标准库来直接使用。看一个简单的例String和DateTime

```
from System import DateTime, String
formatStr = String.Format("{0} {1}", "Hello",
    date and time is ", DateTime.Now)
print formatStr
print dir(String)
raw_input("press Enter to exit!")
```

代码输出如下，可以看到在IronPython代码Format方法进行字符串格式化的输出。



### 引入.NET库

在.NET开发中，会经常通过References来引入IronPython项目中，也可以引用并使用.NET

例如，现在有一个Calc的计算类型，通过这个类型，生成了一个CalcLib.dll。

```
namespace CalcLib
{
    public class Calc
    {
        public int Add(int a, int b)
```

## TCP: 环境搭建(9)

### 推荐排行榜

[JavaScript原型\(211\)](#)[pt的执行上下文\(43\)](#)[TCP: 数据传输\(27\)](#)[pt中的this\(23\)](#)[Script的作用域链\(21\)](#)

```
{
    return a + b;
}

public int Sub(int a, int b)
{
    return a - b;
}
}
```



下面看看如何在IronPython项目中使用这个使用"clr"模块来添加.NET引用：

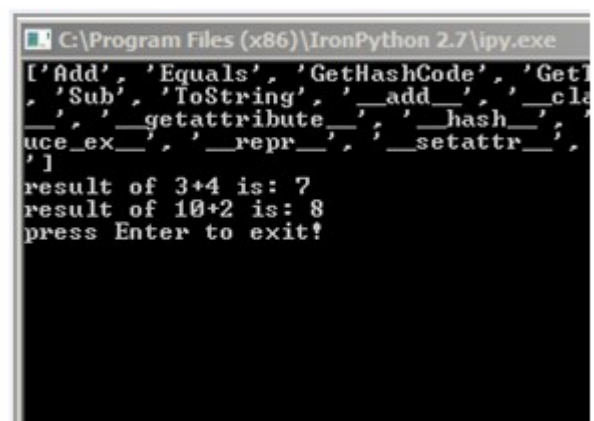


```
import clr
clr.AddReference('CalcLib')
#clr.AddReferenceToFile('CalcLib.dll')
from CalcLib import Calc
print dir(Calc)
calcObj = Calc()
print "result of 3+4 is:", calcObj.Add(3, 4)
print "result of 10+2 is:", calcObj.Sub(10, 2)

raw_input("press Enter to exit!")
```



代码输出如下，当引用了CalcLib.dll之后，创建实例，并且使用实例的C#方法。



### IronPython创建WPF应用

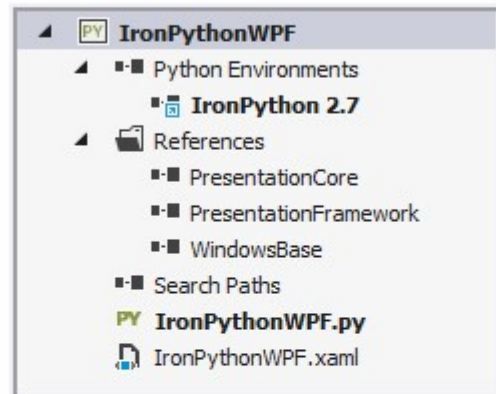
在IronPython项目中，也可以引入WPF相关的库来创建图形界面应用。

安装过PTVS之后，里面有个"IronPython WPF"项目。



过这个模板，可以直接创建WPF应用。

在新建的项目中，VS会帮我们自动引用WPF的个.py和.xaml文件。



下面看一个简单的例子，通过IronPython实现的代码如下：

```
<Grid>
  <Grid.RowDefinitions>
    <RowDefinition></RowDefinition>
    <RowDefinition></RowDefinition>
    <RowDefinition></RowDefinition>
    <RowDefinition></RowDefinition>
    <RowDefinition></RowDefinition>
    <RowDefinition></RowDefinition>
  </Grid.RowDefinitions>
  <Grid.ColumnDefinitions>
    <ColumnDefinition></ColumnDefinition>
    <ColumnDefinition></ColumnDefinition>
    <ColumnDefinition></ColumnDefinition>
    <ColumnDefinition></ColumnDefinition>
  </Grid.ColumnDefinitions>

  <TextBlock Name="InputTb" Grid.Row="0" Grid.Column="0" />
  <TextBlock Name="ResultTb" Grid.Row="0" Grid.Column="1" />

  <Button Content="1" Grid.Row="1" Grid.Column="0" Click="Input_Button_Click"/>
  <Button Content="2" Grid.Row="1" Grid.Column="1" Click="Input_Button_Click"/>
  <Button Content="3" Grid.Row="1" Grid.Column="2" Click="Input_Button_Click"/>
  <Button Content="4" Grid.Row="2" Grid.Column="0" Click="Input_Button_Click"/>
  <Button Content="5" Grid.Row="2" Grid.Column="1" Click="Input_Button_Click"/>
  <Button Content="6" Grid.Row="2" Grid.Column="2" Click="Input_Button_Click"/>
  <Button Content="7" Grid.Row="3" Grid.Column="0" Click="Input_Button_Click"/>
```

```
Click="Input_Button_Click"/>
    <Button Content="8" Grid.Row="4" Grid
Click="Input_Button_Click"/>
    <Button Content="9" Grid.Row="4" Grid
Click="Input_Button_Click"/>
    <Button Content="0" Grid.Row="5" Grid
Click="Input_Button_Click"/>
    <Button Content="+" Grid.Row="2" Grid
Click="Input_Button_Click"/>
    <Button Content="-" Grid.Row="3" Grid
Click="Input_Button_Click"/>
    <Button Content="*" Grid.Row="4" Grid
Click="Input_Button_Click"/>
    <Button Content="/" Grid.Row="5" Grid
Click="Input_Button_Click"/>
    <Button Content="." Grid.Row="5" Grid
Click="Input_Button_Click"/>

    <Button Content="C" Grid.Row="5" Grid
Click="Clear_Button_Click"/>

    <Button Content="=" Grid.Row="1" Grid
Click="Calc_Button_Click"/>
</Grid>
```



对应的IronPython代码如下：



```
from __future__ import division
import traceback
import wpf

from System.Windows import Application, W

class MyWindow(Window):
    def __init__(self):
        wpf.LoadComponent(self, 'IronPyth

    def Calc_Button_Click(self, sender, e
    try:
        result = eval(self.InputTb.Te
        self.ResultTb.Text = str(resu
    except Exception, e:
        tracelog = traceback.format_e
        MessageBox.Show(str(e))

    pass

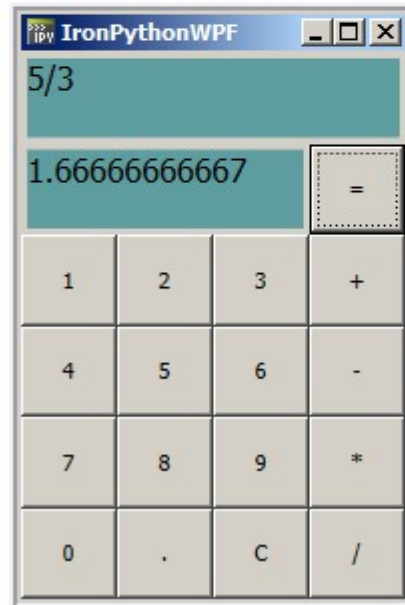
    def Clear_Button_Click(self, sender,
        self.InputTb.Text = ""
        self.ResultTb.Text = ""
    pass
```

```
def Input_Button_Click(self, sender,
                        e):
    self.InputTb.Text += sender.Content
    pass

if __name__ == '__main__':
    Application().Run(MyWindow())
```



代码运行效果如下：



## C#调用IronPython

前面介绍了在IronPython中如何使用.NET库运行IronPython脚本。在.NET framework中，有执行引擎，所以我们可以通过C#代码创建脚本。


先看看我们需要使用的类型：

- ScriptEngine：动态语言（IronPython）；动态语言代码。
- ScriptScope：构建一个执行上下文，其；宿主(Host)可以通过创建不同的 ScriptScope 离的执行上下文。
- ScriptSource：操控动态语言代码的类型；运行（Execute）代码。

现在我们有一个简单的打印当前时间的IronPython脚本：

```
import datetime
print "current datetime is:", datetime.datetime.now()
```

然后就可以使用下面的方式执行脚本：




```
static void Main(string[] args)
{
    try
    {
        ScriptEngine engine = Python.CreateScriptEngine();
        ScriptScope scope = engine.CreateScope();

        ScriptSource script =
engine.CreateScriptSourceFromFile(@"ScriptSource.cs");

        var result = script.Execute(scope);
    }
    catch (Exception e)
    {
        Console.WriteLine(e.Message);
    }

    Console.Read();
}
```




### 给IronPython传递参数

在ScriptScope类型中，有一个SetVariable方法给脚本传递参数。

```
public void SetVariable(string name, object value)
```

这样，我们就可以把一个C#实例传递给IronPython，然后在脚本中使用C#实例的成员。看一个例子：



```
public class Student
{
    public int Age { get; set; }
    public string Name { get; set; }
    public override string ToString()
    {
        return string.Format("{0} is {1} years old",
this.Name, this.Age);
    }
}

class Program
{
    static void Main(string[] args)
    {
        try
```

```
{
    ScriptEngine engine = Python.
    ScriptScope scope = engine.Cr
    Student stu = new Student { N
};

    scope.SetVariable("stuObj", s
    ScriptSource script =
engine.CreateScriptSourceFromFile(@"Print

    var result = script.Execute(s

    }
    catch (Exception e)
    {
        Console.WriteLine(e.Message);
    }

    Console.Read();
}
}
```



在这个例子中，C#代码中创建了一个Student实例传递给了PrintStuInfo.py脚本。

```
print "Student name:", stuObj.Name
print "Student age:", stuObj.Age
print stuObj.ToString()
```

通过输出可以看到，IronPython脚本可以方

```
file:///C:/Workspace/VS/IronPythonTest/CSha
Student name: Wilber
Student age: 28
Wilber is 28 years old
```

## 总结

本篇文章通过一些例子演示了IronPython与C#交互，感觉还是很有意思的。

有时候使用C#调用IronPython可以使程序变的更灵活，C#类型提供一组封装好的操作，每次构建一个操作；这样，用户就可以编写IronPython脚本，实现不同的自定义操作。



作者：[田小计划](#)  
出处：<http://www.cnblogs.com/wilber2013/p/4491...>  
本文版权归作者和博客园共有  
同意必须保留此段声明，且在  
文连接，否则保留追究法律责  
如果觉得不错，请点击[推荐和](#)

分类：[IronPython](#)

好文要顶

关注我

收藏该文



[田小计划](#)  
[关注 - 44](#)  
[粉丝 - 434](#)

[+加关注](#)

« 上一篇: [C#异步委托](#)

» 下一篇: [给博客添加5个的目录:43](#) [田小计划](#) [评论](#)

## 评论列表

[#1楼](#) 2015-05-10 11:02 [史蒂芬King](#)  
支持一下，不错。

[#2楼](#)[楼主] 2015-05-11 09:56 [田小计划](#)  
[@ 史蒂芬King](#)  
多谢支持

[#3楼](#) 2017-01-04 02:09 [太廖](#)  
这个真不错，C#写框架，然后用ironPython 做  
的项目应该很有帮助

注册用户登录后才能发表评论，请[注册](#)  
[页](#)。

[【推荐】50万行VC++源码:大型组态工控库](#)

[【活动】阿里云双11活动开始预热 云服务](#)

[【调查】有奖调研即刻参与,你竟然是酱](#)

[【推荐】Vue.js 2.x 快速入门,大量高效实](#)

[【推荐】腾讯云 普惠云计算 0门槛体验](#)



---

Copyright ©2017 田小计划