Bus\_rode 插件开发

CHMOSGroup |

插件开发

Tad Wiliam

2015

概述与目录

Bus\_rode是一个能够展示一个城市所拥有公交车的状况的一个程序，但这又给我们的程序提出了一个新的要求，能够接驳实时资源，这样才能够使人们的出行更加方便。故，我们开发了一个新的功能模块，用以快速访问实时资源并呈现在bus\_rode之中，这个模块使用.NET的反射机制，实时加载由资源开发者编写的托管类库，获取资源。这需要开发者有足够的编程基础，而且至少会一门.NET语言（C# [即C Sharp] 或 Visual Basic .NET），如果您还未准备好，请先不要看此文，当您觉得时机成熟之时，再来这里，我将会向您阐述如何编写一个可适用于bus\_rode获取实时资源的托管类库dll，下面，我们就开始介绍吧。

目录

[概述与目录 1](#_Toc437702418)

[系统要求 2](#_Toc437702419)

[基本项目构建 3](#_Toc437702420)

[代码解析 5](#_Toc437702421)

[编译配置 10](#_Toc437702422)

[文本规则 12](#_Toc437702423)

[基本语法 12](#_Toc437702424)

[车模块格式 12](#_Toc437702425)

[提交与发布 14](#_Toc437702426)

[结束语 15](#_Toc437702427)

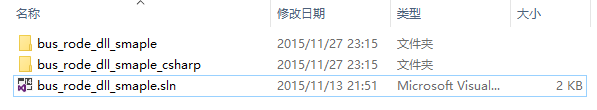
系统要求

1、Visual Studio 2015或更高的版本，版本类别（社区版、专业版…诸如此类的叫做版本类别）不限

2、Windows 7或更高的操作系统，推荐Windows 10

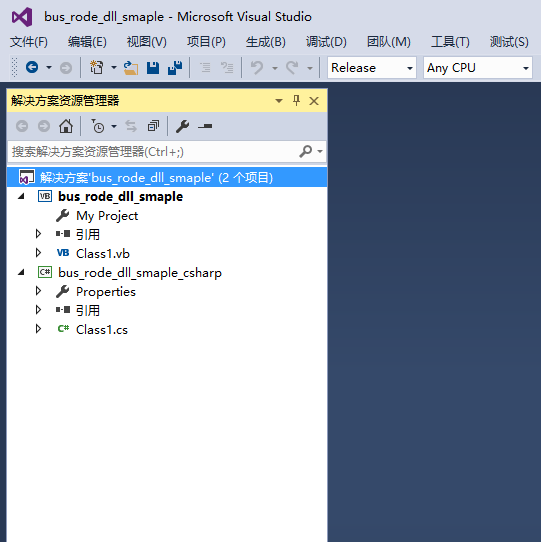
基本项目构建

本文档是bus\_rode模块开发模板的一部分，您可以在本文档所在目录看到以下文件结构：



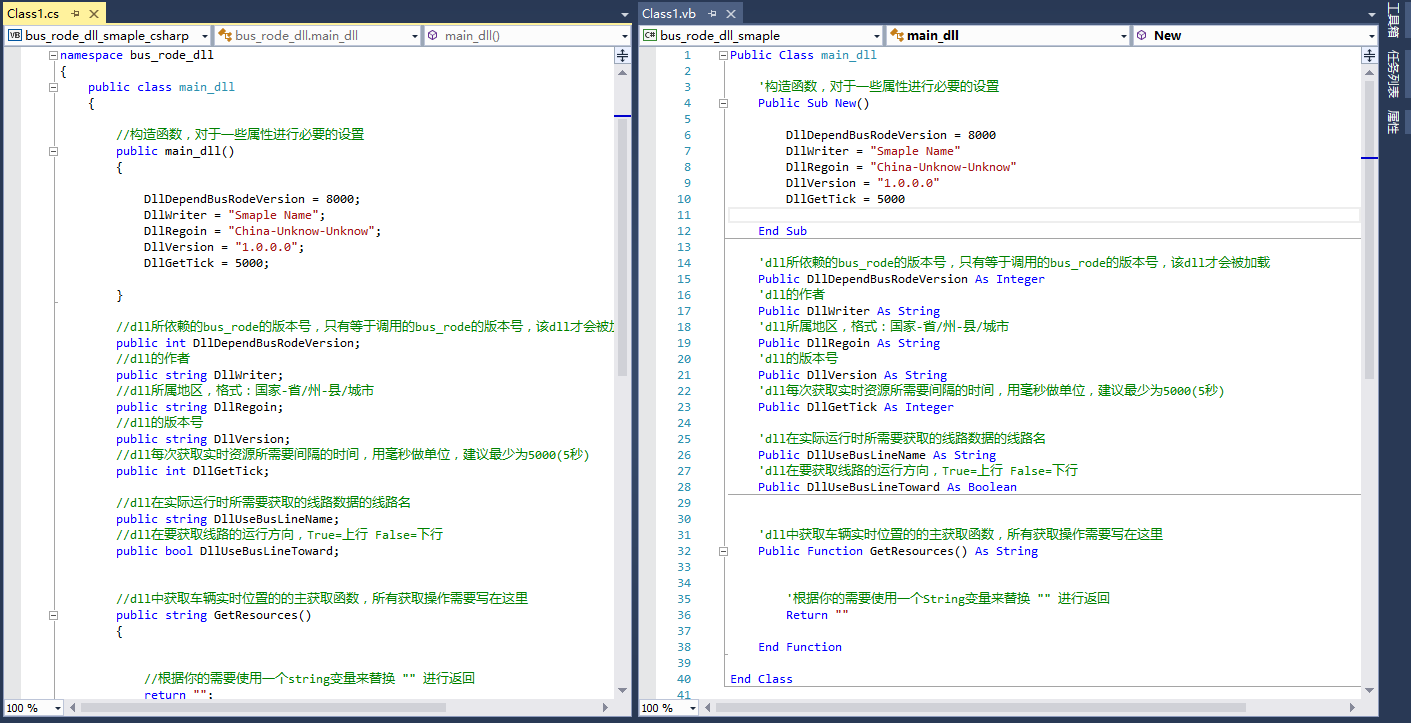
这三者均是模板项目的组成部分，当您打算创建一个模块时，请将这三者复制，在复制的文件上进行修改，不要直接在模板项目上直接修改，以防下次写其他模块时没有模板可用。

双击bus\_rode\_dll\_smaple.sln启动Visual Studio并加载项目后，您可以在解决方案资源管理器中看到项目的结构如下：



其中包含有一个Visual Basic .NET（后文简称VB）项目，和一个C#项目，您不需要两者都写入代码，请选择您最熟悉的一门语言，在其对应的代码中编辑即可。

Class1.cs和Class1.vb分别是C#项目和VB项目的代码文件，所有编写工作将在这里完成。以下展示两者打开的大致内容：



当您在书写完毕后，希望生成dll时，请将你选择的那门语言所对应项目的生成模式调为Release，生成即可，至于如何找到生成的文件，我认为凭您的经验不会找不到的，

代码解析

**头部**

**C#**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace bus\_rode\_dll

{

public class main\_dll

{

**VB**

Public Class main\_dll

此部分代码包含类名、命名空间名等，这些是不容许更改的，C#代码中的using若有需要，可以添加对应的using指令，但只限using .NET中的内建类或命名空间，不容许引用外部类库（第三方类库）

**构造函数**

**C#**

//构造函数，对于一些属性进行必要的设置

public main\_dll()

{

DllDependBusRodeVersion = 8000;

DllWriter = "Smaple Name";

DllRegoin = "China-Unknow-Unknow";

DllVersion = "1.0.0.0";

DllGetTick = 5000;

}

**VB**

'构造函数，对于一些属性进行必要的设置

Public Sub New()

DllDependBusRodeVersion = 8000

DllWriter = "Smaple Name"

DllRegoin = "China-Unknow-Unknow"

DllVersion = "1.0.0.0"

DllGetTick = 5000

End Sub

此部分代码包含在初始化模块时所需要填写的信息，这些信息将会显示在模块信息中。

DllDependBusRodeVersion表示支持的bus\_rode的build数字，我们需要保证模块能够正确地被加载在正确的bus\_rode上，所以该数字务必与该模块所支持的bus\_rode版本号相同，在版本号不同时，模块不会被加载，只能看到模块信息。

DllWriter表示编写者的姓名。

DllRegoin表示该模块获取的是那个地方的实时信息，该文本应当与资源文件readme.txt中的第一行（注意，不是第二行），即资源所在地址的文本必须相同，若不相同，模块也不会被加载。至于该如何填写，请参阅资源开发手册readme.txt有关部分。

DllVersion表示模块的版本号。

DllGetTick表示模块每获取一次资源间隔的时间，以 **毫秒** 作为单位（1000毫秒=1秒）。尽量为500的倍数，当不为500的倍数的时候，会以Int( DllGetTick / 500 )取整。建议最小值为5000（5秒）。

如果您在这个类中添加了其余内容，而且需要在初始化时写入一些值，可以依据您的需要增加更多语句，但以上的5个赋值语句是必不可少的，不得删去。

**变量定义**

**C#**

//dll所依赖的bus\_rode的版本号，只有等于调用的bus\_rode的版本号，该dll才会被加载

public int DllDependBusRodeVersion;

//dll的作者

public string DllWriter;

//dll所属地区，格式：国家-省/州-县/城市

public string DllRegoin;

//dll的版本号

public string DllVersion;

//dll每次获取实时资源所需要间隔的时间，用毫秒做单位，建议最少为5000(5秒)

public int DllGetTick;

//dll在实际运行时所需要获取的线路数据的线路名

public string DllUseBusLineName;

**VB**

'dll所依赖的bus\_rode的版本号，只有等于调用的bus\_rode的版本号，该dll才会被加载

Public DllDependBusRodeVersion As Integer

'dll的作者

Public DllWriter As String

'dll所属地区，格式：国家-省/州-县/城市

Public DllRegoin As String

'dll的版本号

Public DllVersion As String

'dll每次获取实时资源所需要间隔的时间，用毫秒做单位，建议最少为5000(5秒)

Public DllGetTick As Integer

'dll在实际运行时所需要获取的线路数据的线路名

Public DllUseBusLineName As String

DllDependBusRodeVersion、DllWriter、DllRegoin、DllVersion和DllGetTick在前面已介绍过对应作用，这里的语句只是用来声明的，不允许修改。

我们来关注一下剩余的2项。

DllUseBusLineName表示当前需要返回实时信息的线路名称，该名称的来源是资源文件have\_bus.txt中的线路名，至于具体的名称格式，可以参阅资源开发手册have\_bus.txt一节。该变量由bus\_rode每一次调用时实时修改，所以在主函数（见后文）中，需要实时关注该变量，使用该变量中指定的线路名来获取实时资源。该变量不得在任何情况下非法修改，只能在bus\_rode的调用下由bus\_rode自身修改。

获取线路时还需要注意的是需要同时获取上行的和下行的。然后一并返回。

如果您还需要定义一些其余全局变量，可以补充更多语句，但上面的语句都是必须的，不可修改。

**主函数**

**C#**

//dll中获取车辆实时位置的的主获取函数，所有获取操作需要写在这里

public string GetResources()

{

//根据你的需要使用一个string变量来替换 "" 进行返回

return "";

}

**VB**

'dll中获取车辆实时位置的的主获取函数，所有获取操作需要写在这里

Public Function GetResources() As String

'根据你的需要使用一个String变量来替换 "" 进行返回

Return ""

End Function

该函数是每间隔DllGetTick由bus\_rode调用一次，这个函数也是获取的核心部分，该函数在bus\_rode内部已作了多线程处理，无须担心过多的获取会卡死bus\_rode。

Return所返回的是一个按一定规则组织的字符串，在后文我们会介绍如何遵守这个规则，Return之后的 ””应当替换为一个有效的String变量，该变量存储了按规则写好的字符串，可以传回空字符串，表示获取不到资源。

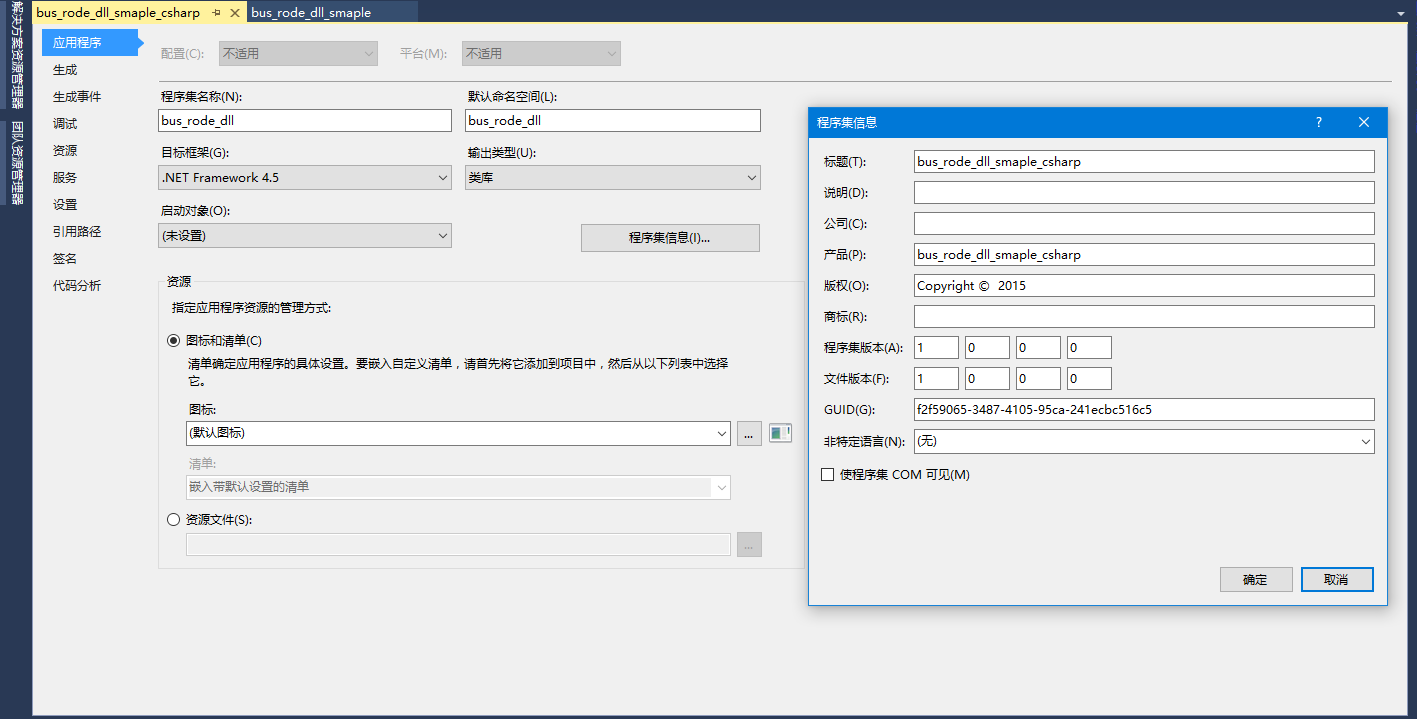
如果对源码的编写仍有疑惑，可以去翻一翻bus\_rode的源代码，相关获取代码在system.vb和MainWindow.xaml.vb中的计时器部分，或许会让你的理解更深一些。

编译配置

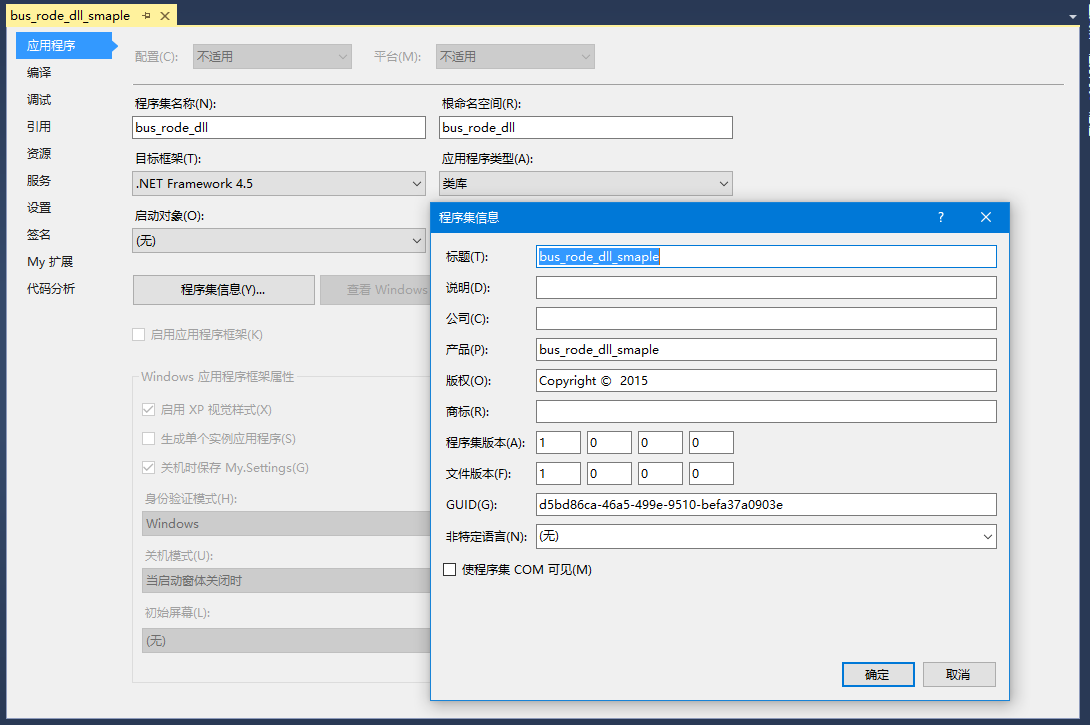
文件的编译的相关配置我们已经在模板中配置完毕了，请勿擅自修改。

以下展示相关的设置图：

C#



VB



程序集名称、根命名空间、应用程序类型等等都是配置好的，不能修改，修改后一定无法加载插件。

目标框架是与bus\_rode的框架保持一致的，这里，bus\_rode采用的框架为.NET 4.5，也不要修改。如果以后依赖的.NET版本改变了，模板会更新，请使用最新的模板创建模块。

但是，程序集信息这一项是可以修改的，其中的标题、说明等都可以根据您的需要修改，该部分内容不会被bus\_rode读取，只是会显示在 文件资源管理器 之中。

文本规则

本部分主要介绍返回文本的格式，返回的文本必须遵守该格式，否则不会被识别。

基本语法

*车模块1@车模块2@车模块3@车模块4@......@车模块n*

其中@为分隔符，分隔每个模块，这部分在程序中使用Split(“@”)来实现，如果您懂这一点，就非常好理解了。

车模块格式

*LineName#Toward#StationName#Message1#Message2#Message3*

其中#为分隔符，分隔每个数据，这部分在程序中使用Split(“#”)来实现，如果您懂这一点，就非常好理解了。

其中，LineName表示要描述的线路名，必须是资源文件have\_bus.txt中有的线路名。Toward的合法字段为 上行 或 下行 ，他表示这辆车在线路中运行的方向，至于具体的上下行，须符合bus.txt中对于上下行的定义。

StationName意义是当前车到达过的最近的一个站台，而不是下一站，这点需要注意，这个部分的文字必须是bus.txt中该线路有的站台名称。这个部分**不仅**可以填入当前车到达过的最近的一个站台**还可以**填入这个车经过的站台数（后文称为 PassStation），二者选其一填写即可。PassStation的数值将会被填写入mid\_bus数组中并显示在界面中，请确保这个数字是基于0起始的，而且不要大于200，这个数字对应在线路界面中站台列表的item序号，如果您还不能理解PassStation，可以查看bus\_rode的源码。

Message1、2、3分别是传输时需要附加的说明，可以为空，即：*LineName#Toward#StationName###*

提交与发布

当测试稳定之后，您可以将整个项目打包交付CHMOSGroup，由我们查看代码是否合格，最后在合格的情况下授权发布这个模块，给公众使用。

该模块不得包含任何恶意代码，也不得有任何广告嫌疑，由于我们采用了dll作为模块加载的方式，这很容易成为攻击者的目标，所以我们也会对dll严加检查，防止任何不符合规范的dll流入用户的手中。

结束语

模块开发文档至此已结束，如果有什么疑问，可以多翻源码或者看看资源开发手册，希望您能跟我们做的更好！

文章写得匆忙，未尽到之处望见谅，以后会加以补充。

再见！

Saturday, December 12, 2015