目录

[简介 1](#_Toc353282916)

[环境 1](#_Toc353282917)

[目的 1](#_Toc353282918)

[框架文件结构 2](#_Toc353282919)

[业务文件结构 2](#_Toc353282920)

[部署 3](#_Toc353282921)

[配置文件 3](#_Toc353282922)

[本地配置 3](#_Toc353282923)

[IIS配置 5](#_Toc353282924)

[服务寄宿 6](#_Toc353282925)

[本地宿主/Windows 服务 6](#_Toc353282926)

[IIS寄宿 6](#_Toc353282927)

[文件结构 7](#_Toc353282928)

[框架实现 7](#_Toc353282929)

[流程图 7](#_Toc353282930)

[客户端请求数据格式 8](#_Toc353282931)

[服务器返回数据格式 8](#_Toc353282932)

[服务器流程 9](#_Toc353282933)

[初始化 9](#_Toc353282934)

[URI 9](#_Toc353282935)

[数据解析 9](#_Toc353282936)

[版本 10](#_Toc353282937)

[版本号 10](#_Toc353282938)

[1.0.0.0 10](#_Toc353282939)

[1.0.0.1 10](#_Toc353282940)

# 简介

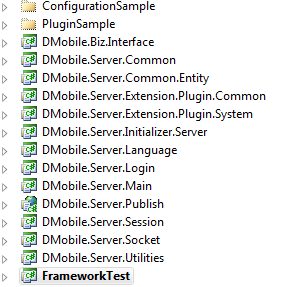
## 环境

基于.NET Framework 4.0,仅适用于4.0及以上环境。

## 目的

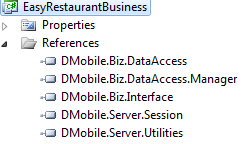
DMobile.Framework目的是为了方便iOS、Android、Win8等移动客户端与服务器交互的轻量级服务器框架。

## 框架文件结构



1. DMobile.Biz为业务接口，开发人员使用的业务必须继承DMobile.Biz.Interface。
2. DMobile.Server为服务器通讯、扩展插件、发布、会话等方法，开发人员扩展插件、指定服务器通讯内容解析方法应继承相应接口实现。
3. Sample展示了如何继承并实现。

## 业务文件结构

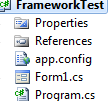


这是一个业务示例，业务应继承DMobile.Biz.Interface

# 部署

## 配置文件

### 本地配置



配置程序配置文件，加入服务器节点，和服务器配置

<configuration>

<startup>

<supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.0"/>

</startup>

<system.web>

<httpRuntime maxRequestLength="65535"/>

</system.web>

<system.serviceModel>

<standardEndpoints>

<webHttpEndpoint>

<standardEndpoint name="LoginEndPoint" maxBufferSize="2147483647"

maxBufferPoolSize="2147483647" maxReceivedMessageSize="2147483647"

defaultOutgoingResponseFormat="Json" helpEnabled="true">

<readerQuotas maxDepth="2147483647" maxStringContentLength="2147483647"

maxArrayLength="2147483647" maxBytesPerRead="2147483647" maxNameTableCharCount="2147483647" />

</standardEndpoint>

</webHttpEndpoint>

</standardEndpoints>

<behaviors>

<serviceBehaviors>

<behavior name="LoginServiceBehavior">

<bufferedReceive maxPendingMessagesPerChannel="1024" />

<serviceThrottling />

</behavior>

</serviceBehaviors>

</behaviors>

<bindings>

<webHttpBinding>

<binding name="LoginBinding" maxBufferSize="2147483647" maxBufferPoolSize="2147483647"

maxReceivedMessageSize="2147483647">

<readerQuotas maxDepth="2147483647" maxStringContentLength="2147483647"

maxArrayLength="2147483647" maxBytesPerRead="2147483647" maxNameTableCharCount="2147483647" />

</binding>

<binding name="CommunicationBinding" maxBufferSize="2147483647"

maxBufferPoolSize="2147483647" maxReceivedMessageSize="2147483647">

<readerQuotas maxDepth="2147483647" maxStringContentLength="2147483647"

maxArrayLength="2147483647" maxBytesPerRead="2147483647" maxNameTableCharCount="2147483647" />

</binding>

</webHttpBinding>

</bindings>

<services>

<service name="DMobile.Server.Main.CommunicationCenter">

<endpoint address="http://localhost:4477/CommunicationCenter"

binding="webHttpBinding" bindingConfiguration="CommunicationBinding"

name="basicWeb" contract="DMobile.Server.Main.ICommunicationCenter" />

</service>

<service name="DMobile.Server.Login.LoginService">

<endpoint address="http://localhost:4477/LoginService" binding="webHttpBinding"

bindingConfiguration="LoginBinding" name="Login" contract="DMobile.Server.Login.ILoginService"

kind="webHttpEndpoint" endpointConfiguration="LoginEndPoint" />

</service>

</services>

</system.serviceModel>

<appSettings>

<!--Configer-->

<add key="ServerConfigFile" value="DMobile.Server.Configuration.dll"/>

<add key="BizFile" value="EasyRestaurantBusiness.dll"/>

</appSettings>

</configuration>

1. 注意appSettings的节点路径，其中BizFile不是必须的，但在bin文件夹业务文件下应以Business结尾。ServerConfigFile必须指定服务器配置方法的dll。
2. Endpoint 节点地址可以任意修改，不过要符合地址规则。

### IIS配置

<configuration>

<system.web>

<compilation debug="true" targetFramework="4.0" />

<httpRuntime maxRequestLength="65535" />

</system.web>

<system.webServer>

<modules runAllManagedModulesForAllRequests="true">

<add name="UrlRoutingModule"

type="System.Web.Routing.UrlRoutingModule, System.Web, Version=4.0.0.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a" />

</modules>

</system.webServer>

<system.serviceModel>

<serviceHostingEnvironment aspNetCompatibilityEnabled="true" />

<standardEndpoints>

<webHttpEndpoint>

<!--

Configure the WCF REST service base address via the global.asax.cs file and the default endpoint

via the attributes on the <standardEndpoint> element below

-->

<standardEndpoint name="" helpEnabled="true" automaticFormatSelectionEnabled="true" />

</webHttpEndpoint>

</standardEndpoints>

</system.serviceModel>

<appSettings>

<!--IIS Configer-->

<add key="ServerConfigFile" value="bin\DMobile.Server.Configuration.dll"/>

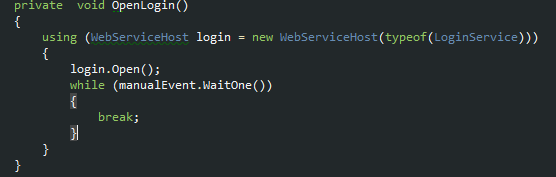
<add key="BizFile" value="bin\EasyRestaurantBusiness.dll"/>

</appSettings>

</configuration>

## 服务寄宿

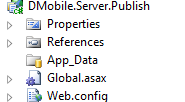
### 本地宿主/Windows 服务



在程序或者服务中，使用以上代码就可以开启服务了。

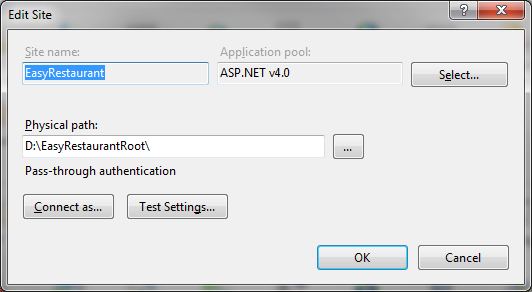
其中manualEvent目的是不让这个线程退出，保持服务运行。

### IIS寄宿



1、右键单击项目，选择发布（publish）

2、选择文件系统，发布至IIS指定的目录

例如：我的IIS指定为D:\EasyRestaurantRoot\,则发布至这个文件夹下，就可以访问了。

细节请看<http://www.cnblogs.com/KeithWang/archive/2012/02/14/2351826.html>

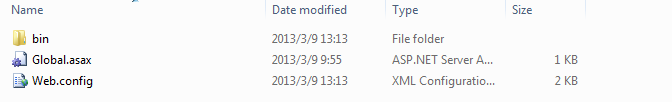
### 文件结构

1. 保证plugin在plugins文件夹下
2. 保证业务文件在bin下

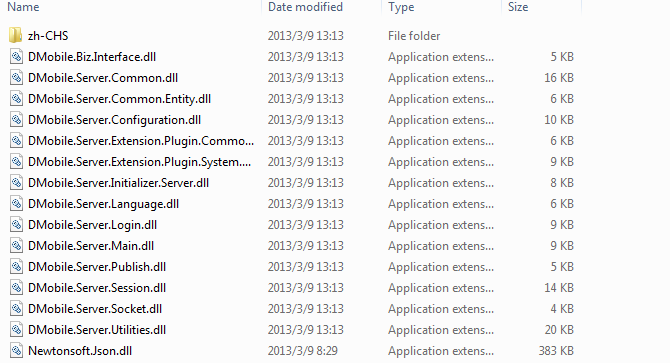
以下为一个例子：

第一级：





Bin：



# 框架实现

## 流程图

## 客户端请求数据格式

验证数据、会话

服务器初始化

解析请求

返回

通知plugin

提交业务层

**框 架**

会话数据

序列化Json，压缩

客户端

序列化为Json的数据

格式统一为Json格式。

请求数据格式分为两部分：

每部分格式都为{“Key”:key,”Vaule”:value}

1. 所请求的方法{“Key”:”METHOD”,”Value”:”方法格式Json”}
2. 所附加的信息{“Key”:”ATTACHMENT”,”Value”:”附加信息Json”}

方法格式与DMobile.Server.Common.MethodSchema一致，附加信息与DMobile.Server.Common. AttachmentSchema一致

例如：

[{"Key":"METHOD","Value":"{"MethodName":"OperationUserLocation","Parameters":{"location":{"UserID":"admin","IPAddress":"127.0.0.1","Port":0,"NetWorkType":0,"Latitude":33.2355,"Longitude":-325.232,"City":30,"OnlineStatus":0},"operationType":"2"}}"},{"Key":"ATTACHMENT","Value":"{"MacAddress":"Empty"}"}]

## 服务器返回数据格式

格式统一为Json格式（未压缩情况下）

例如：

{"location":{"UserID":"admin","IPAddress":"127.0.0.1","Port":0,"NetWorkType":0,"Latitude":33.2355,"Longitude":-325.232,"City":30,"OnlineStatus":0}

## 服务器流程

### 初始化

在服务端（LoginService、CommunicationCenter）构造函数中

PluginManager.Instance.LoadPlugins(PluginDetector.GetPlugins());

MethodRoute.Register(ConfigurationLoader.Instance.Business.GetType());

这两个函数分别加载插件和业务

### URI

**LoginServic**

Get方法：http://节点地址/request?data={encryptData}

POST方法: http://节点地址//post/{dataLength}，传入的流数据为Json数据

**CommunicationCenter**

Get方法：http://节点地址//request?sessionid={sessionId}&encryptdata={encryptData}

POST方法: http://节点地址/post/{dataLength}，传入的流数据为Json数据

### 数据解析

服务器将使用**IAttachmentResolver，IDataParser，IMethodResolver，IRequestSpliter，ISecurityAuthorize**这几个接口对传入的数据进行解析，具体请看DMobile.Server.Configuration，服务器加载此配置作为解析器。

**IDataParser：**数据加密解密，请保持客户端、服务端使用同一种算法加解密

**IMethodResolver：**将对请求的方法解析，以适配业务层的方法（NOTE：一般不重新定义此接口的实现）

**IRequestSpliter：**将对客户端传入的数据解析成各个子字符串以备对应解析方法使用

**ISecurityAuthorize：**扩展方法，安全认证的解析

**IAttachmentResolver：**附加信息解析

# 版本

# 版本号

### 1.0.0.0

作者：Jason(董健) 成都

日期：2013年3月9日

### 1.0.0.1