# 项目介绍

很多手机APP会不定时的给用户推送消息，例如一些新闻APP会给用户推送用户可能感兴趣的新闻，或者APP有更新了，会给用户推送是否选择更新的消息等等，这就是所谓的“消息推送”。此项目主要是基于SuperSocket的消息推送项目，打通了B/S（浏览器/服务器）与C/S（客户端/服务器）的网络通讯。

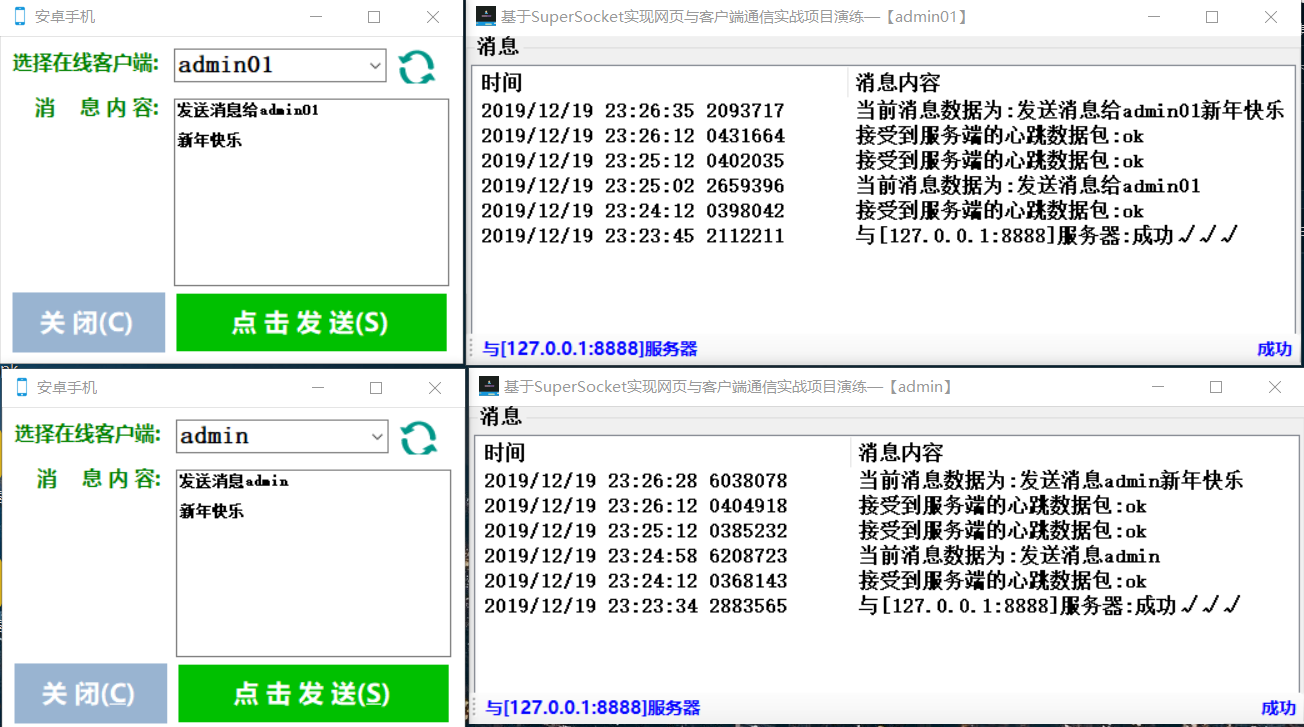
**在项目中：**

1. 我们将GA.SuperSocket.Service项目作为我们的Web服务站点，在其站点下寄宿了Socket服务，当站点启动后，socket服务也会跟着启动；
2. GA.SuperSocket.AppClient是模仿的桌面程序（PC端），实现登录并接收消息
3. GA.SuperSocket.MobileApp是模仿的移动手机端（发送消息功能），对这个客户端（发送消息）可以打开多次，相当于有多部手机，发送消息发给Socket服务，由Socket服务转发消息给指定的客户端（PC端）接收消息。

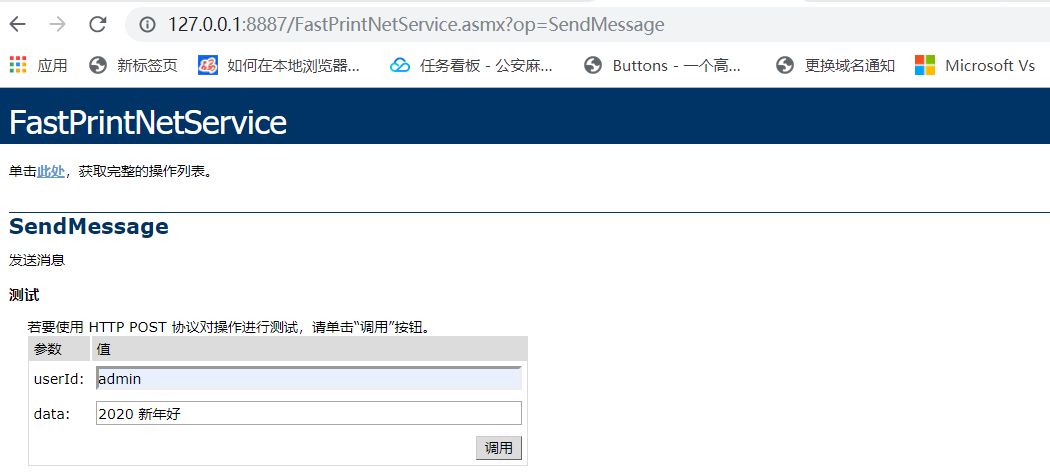
**项目截图：**

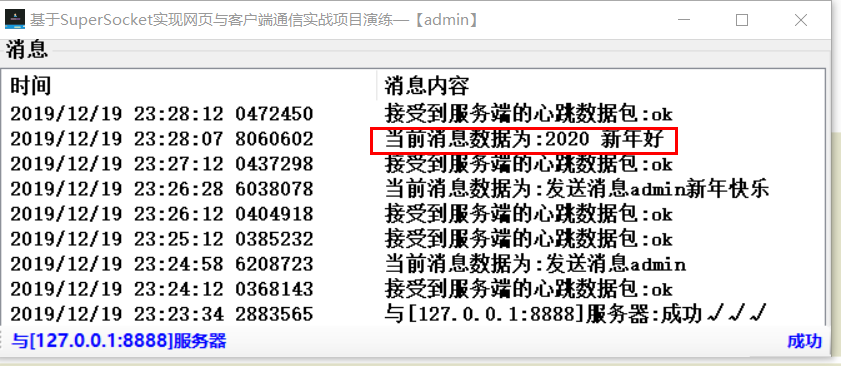
* 多个客户端与服务端的socket服务连接并给指定用户发送消息





* 通过浏览器给指定用户发送消息





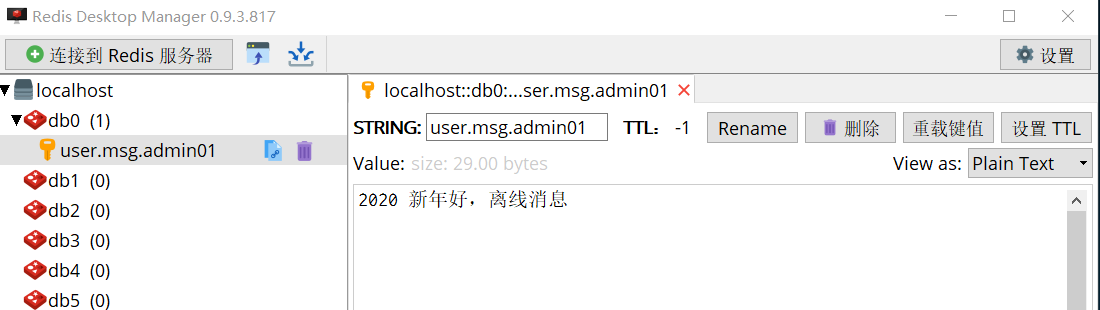
* 发送离线消息给admin01



发送成功后的提示，表示离线消息存入到了redis



我们打开redis客户端查看



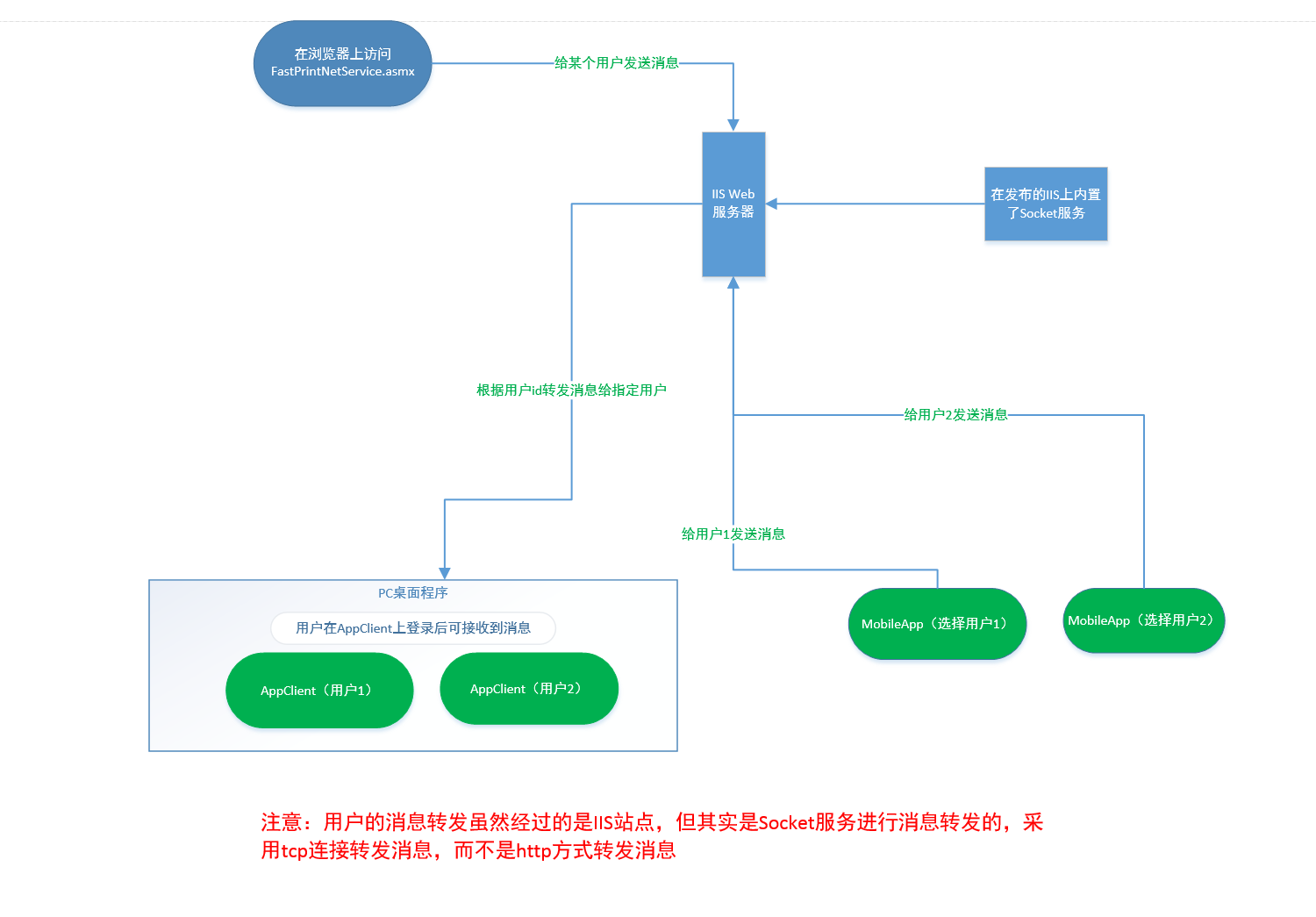
重新登录



收到离线消息



**通信流程图：**



**实现功能如下：**

1. 实现模拟手机端对PC端的桌面程序发送消息，即客户端之间形成网络通讯进行消息发送；
2. 实现服务端发送消息给PC端的桌面程序；
3. 实现客户端掉线后由服务端发送消息后缓存到Redis作为离线消息，当客户端登录上线后立马收到消息；
4. 实现客户端断线重连功能。

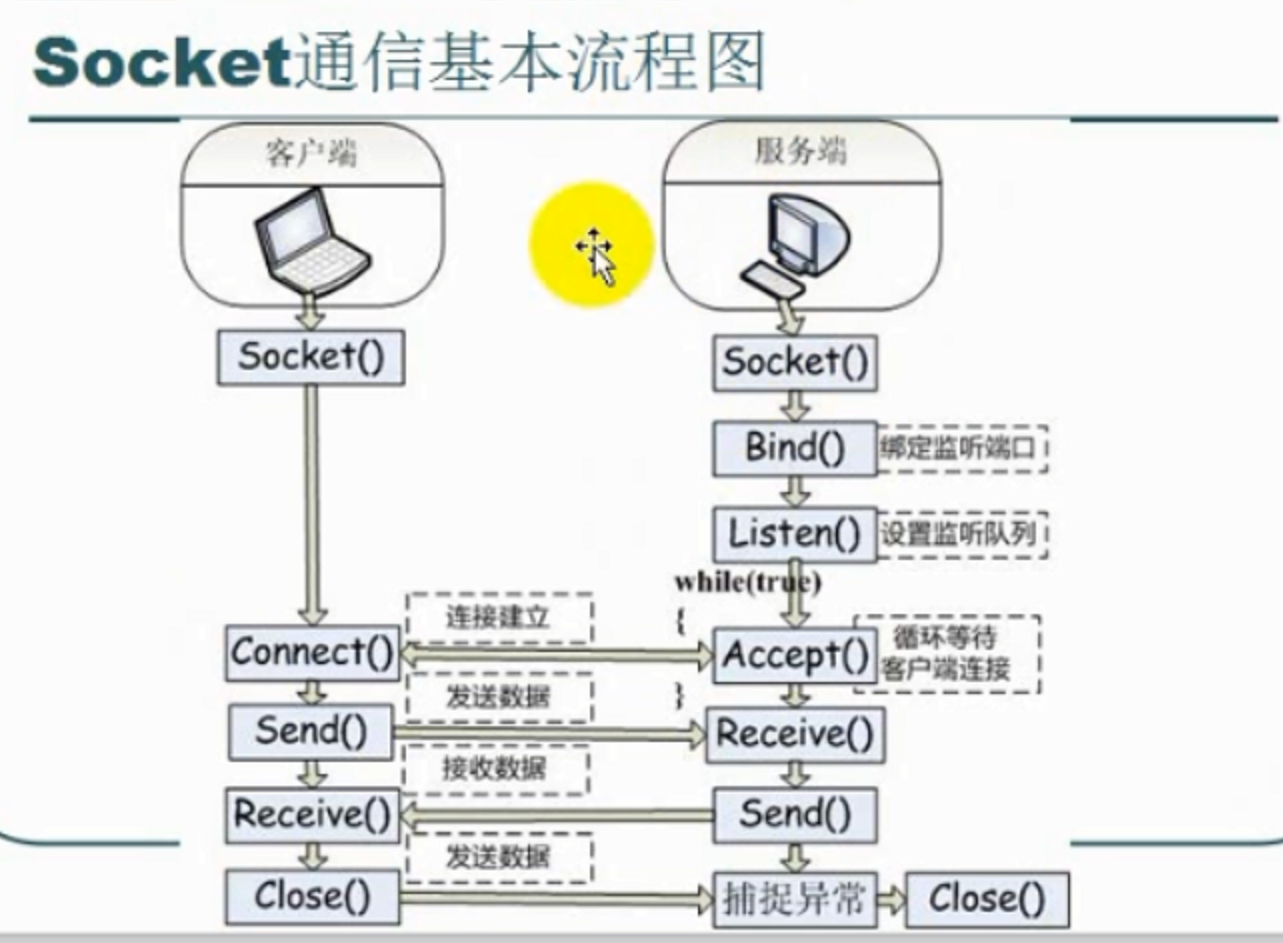
# 知识点覆盖

下面是项目所覆盖的知识点介绍：

1. C# Winform技术，主要是实现客户端的模拟；
2. .net WebService基本使用技巧；
3. .net开源SuperSocket实现服务端向指定客户端推送消息；
4. IIS程序发布与部署；
5. TCP/IP端口号的问题处理以及部署Redis（缓存数据库）的基本操作和使用；
6. 为了防止网络抖动出现TCP假死连接，Socket客户端与服务端互发心跳数据包，维持长连接；
7. 服务端崩溃或重启站点后Socket客户端实现自动断线重连；
8. 防止IIS自动回收应用程序进程的基本设置；
9. Redis消息队列存储用户离线数据实现消息的持久化。

**学习建议：**由于Socket属于高级编程，覆盖的知识点稍微比较多，建议至少拥有C# Winform以及.net mvc的基础初学者可以学习，否则不建议去学。

# Socket通信基本流程图



流程图解读：

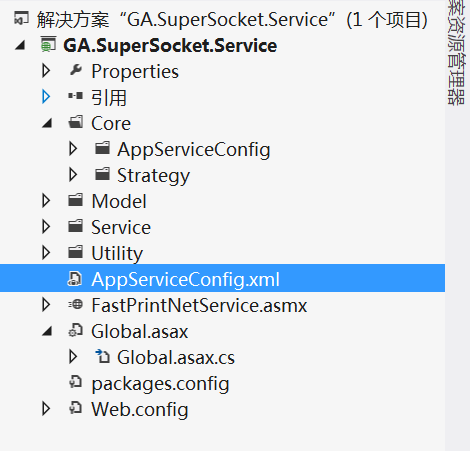
* **服务端初始化：**进行Socket()、Bind()、Listen()监听；
* **客户端：**建立socket()发起对服务端的连接Connect()；
* **服务端：**接受Accept()到连接请求，创建socket会话通信，这里服务端会不断的循环监听等待所有客户端的连接请求；
* **客户端：**连接成功后Send()发送数据告知服务端接收Receive()消息连通好了；
* **服务端：定时发送**Send()心跳数据包给客户端；
* **客户端：**接收Receive()心跳数据包，由服务端维持与客户端的长连接；
* **客户端：**退出会话后关闭Close() socket会话连接；
* **服务端：**关闭该客户端的socket会话通信

# 框架及代码解读

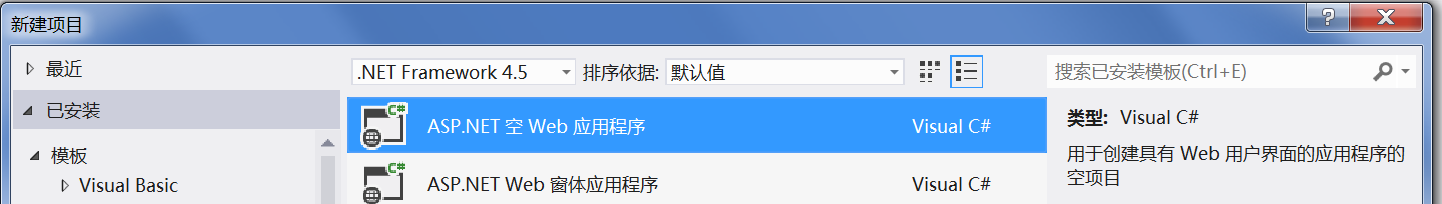
服务端即部署的IIS站点，PC端和移动端均属于不同平台下的客户端。

## 服务端

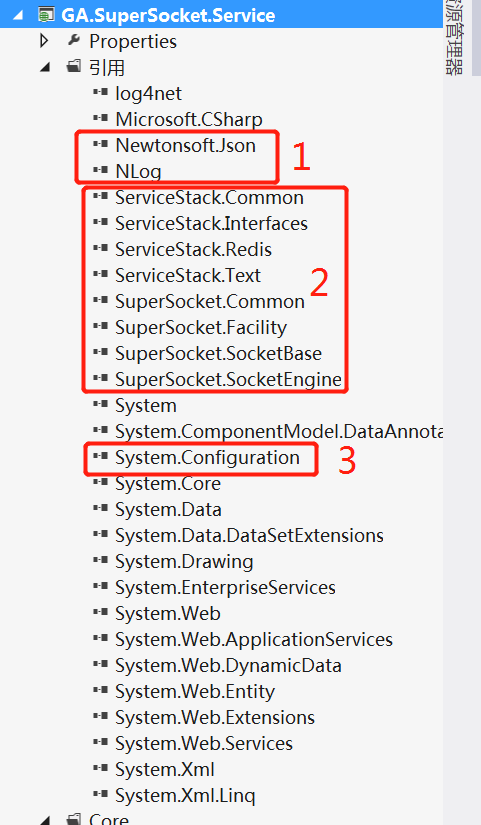
### 框架解读



1. **GA.SuperSocket.Service** 属于Web应用程序，.net framework 4.5版本



1. **引用**



（1）序列化对象和日志组件库;

（2）redis以及socket核心组件库；

（3）读取Web.config的库

1. **Core**

属于项目的核心程序，主要实现了服务端的socket监控、发送心跳数据包维持长连接、接收客户端消息、发送消息、采用redis转发离线消息等功能。

1. **Model**

自定义的实体类

1. **Service**

通过调用Core的核心方法对外提供接口

1. **Utility**

辅助及扩展类

1. **AppServiceConfig.xml**

xml配置文件，配置了redis的连接字符串



adminpwd是连接redis数据库的验证密码，127.0.0.1是本机ip；6379是redis默认的连接运行端口

注意：在使用redis之前请部署好redis，否则在Core中的redis核心程序会报错，而导致无法使用。



1. **FastPrintNetService.asmx**

作为开放给客户端的Web引用文件，主要是提供接口，默认启动程序时在浏览器打开的文件。

1. **Global.asax**

程序启动文件，初始化配置以及启动socket服务程序

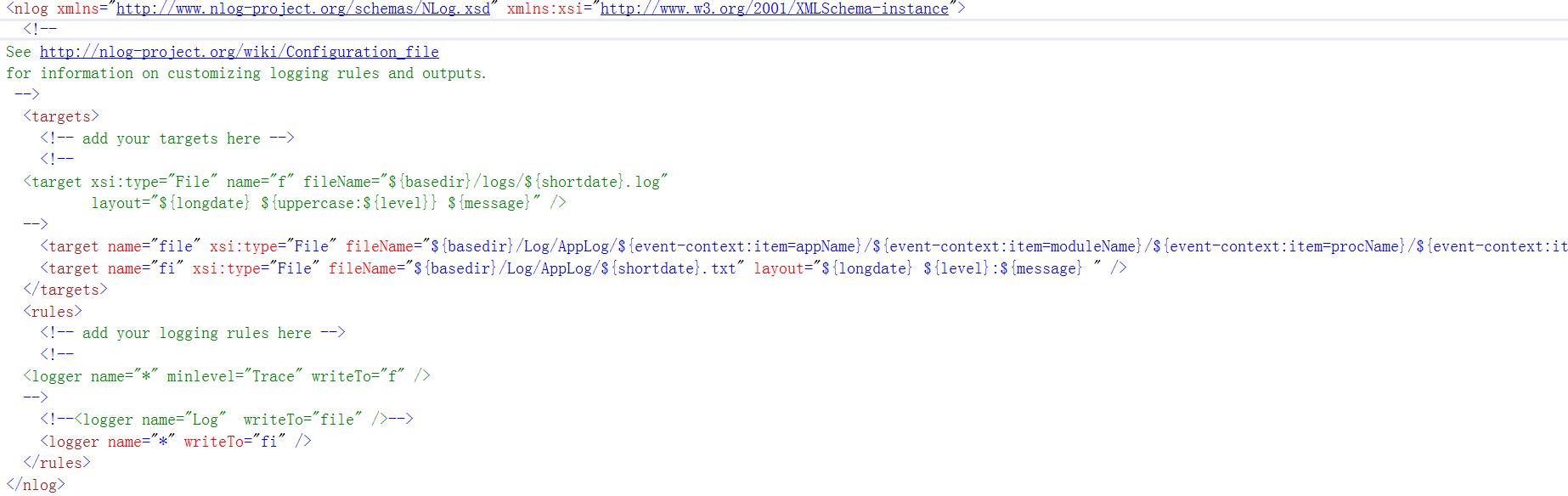
1. **Packages.config**

引用的包文件，包含了各个引用的文件包名、版本、.net framework版本

1. **Web.config**

主要是配置日志信息（日志组件库、打印日志的输出格式）以及socket监听端口





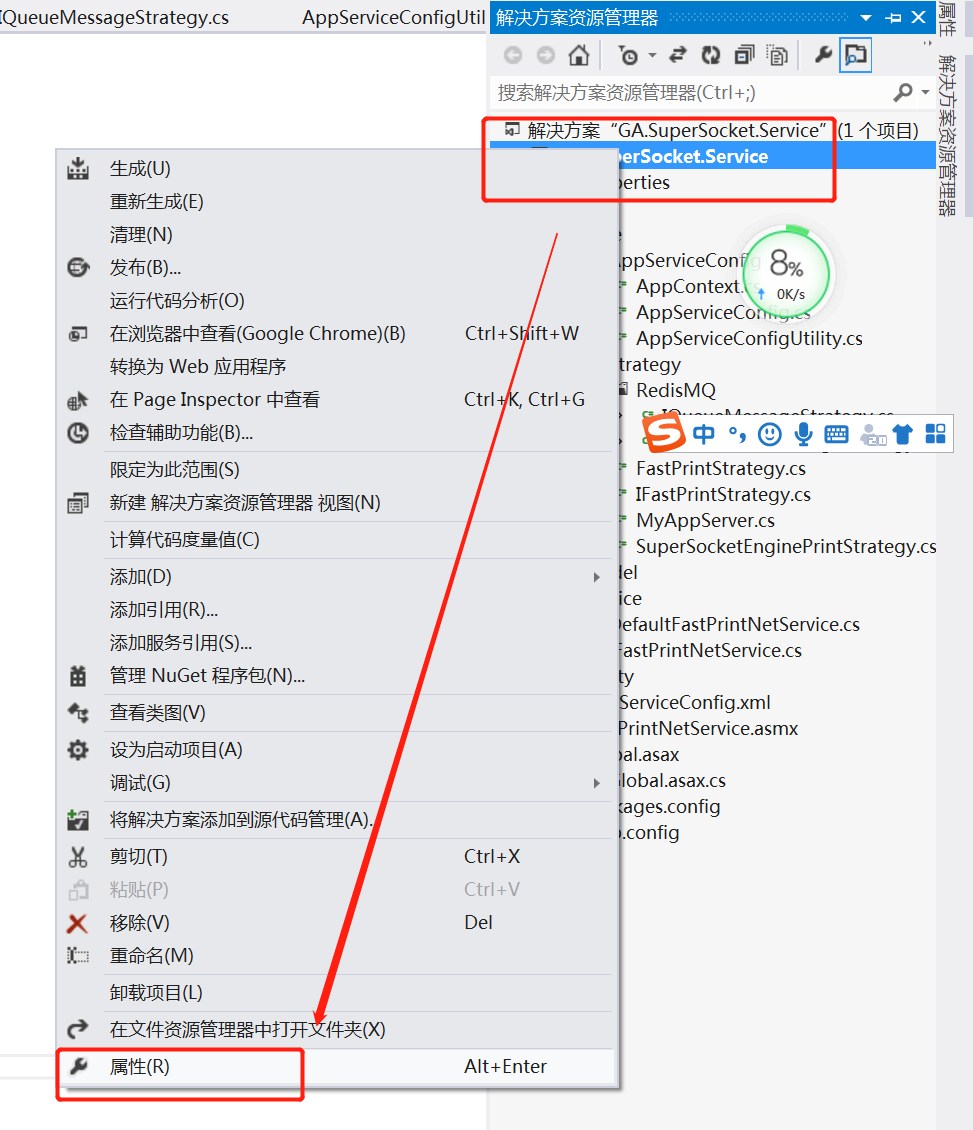
### 初始化配置

* **Socket监听端口**

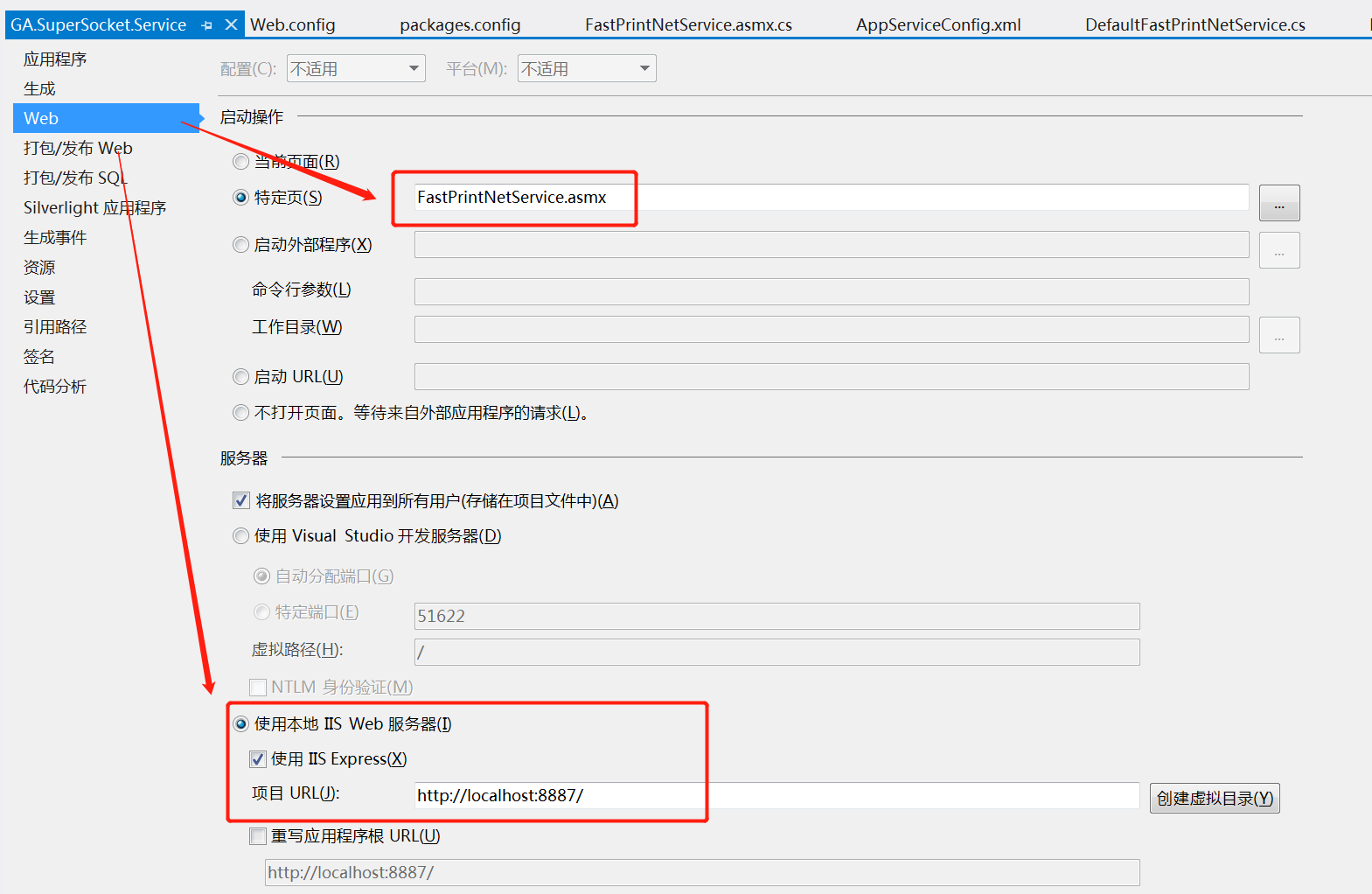


在这里我们配置的监听端口是8888（可自己修改），此端口是socket连接客户端的端口，并非服务程序配置在IIS的端口。

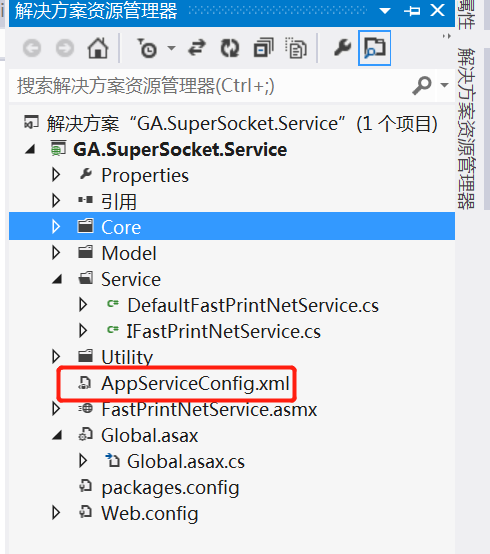
* **启动文件配置**



配置启动时打开的默认页：FastPrintNetService.asmx和程序启动IIS路径，在这里端口号设置为8887（可自己修改）



* **运行时注意**

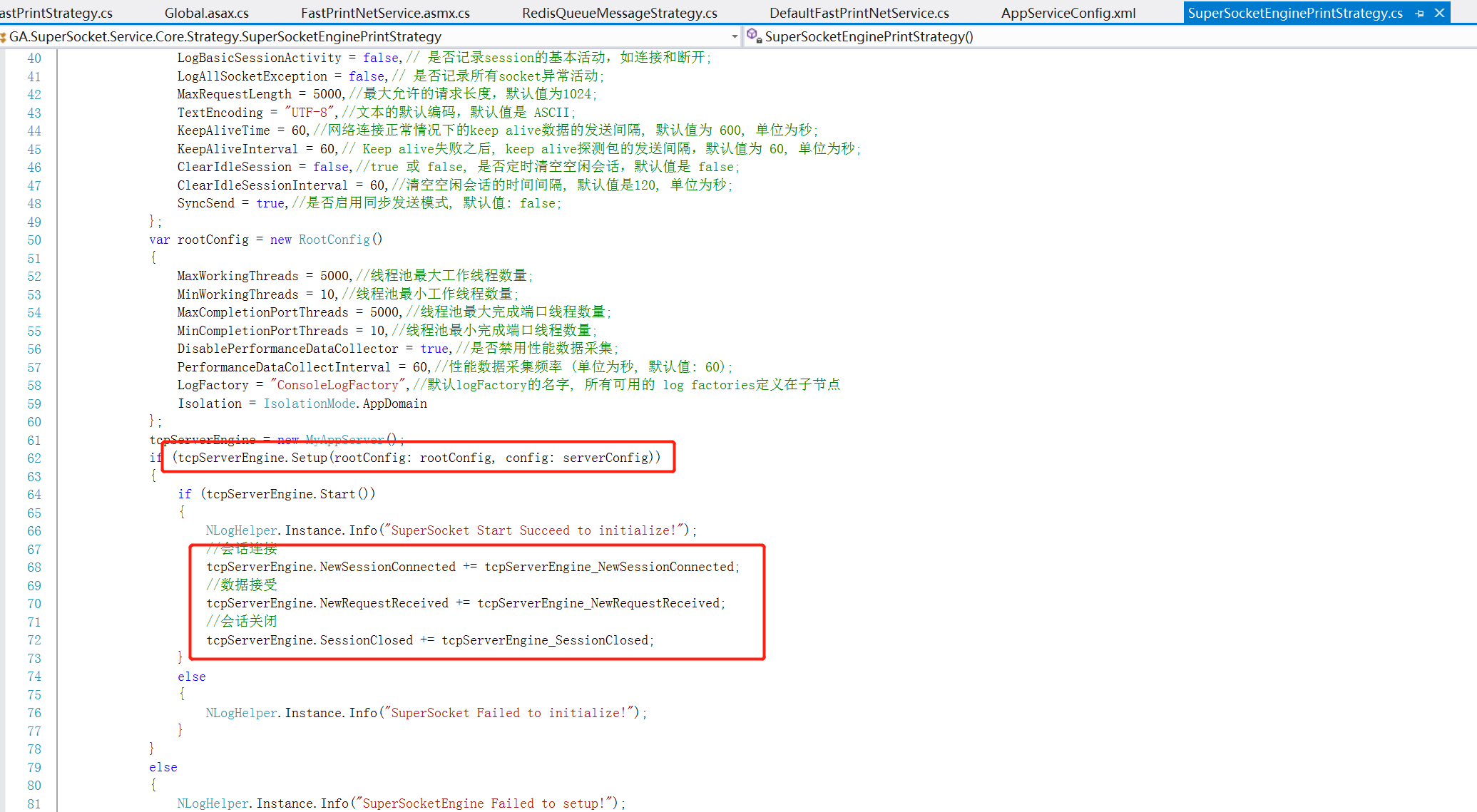


该文件的属性需要设置为 **始终复制**，程序启动编译时会编译到bin文件夹下，否则不会编译进去，程序读取到该文件时就会报错。



### 代码解读

1. 文件名：SuperSocketEnginePrintStrategy.cs是静态文件，不需要new对象就可以直接使用其方法
2. **初始化socket服务**



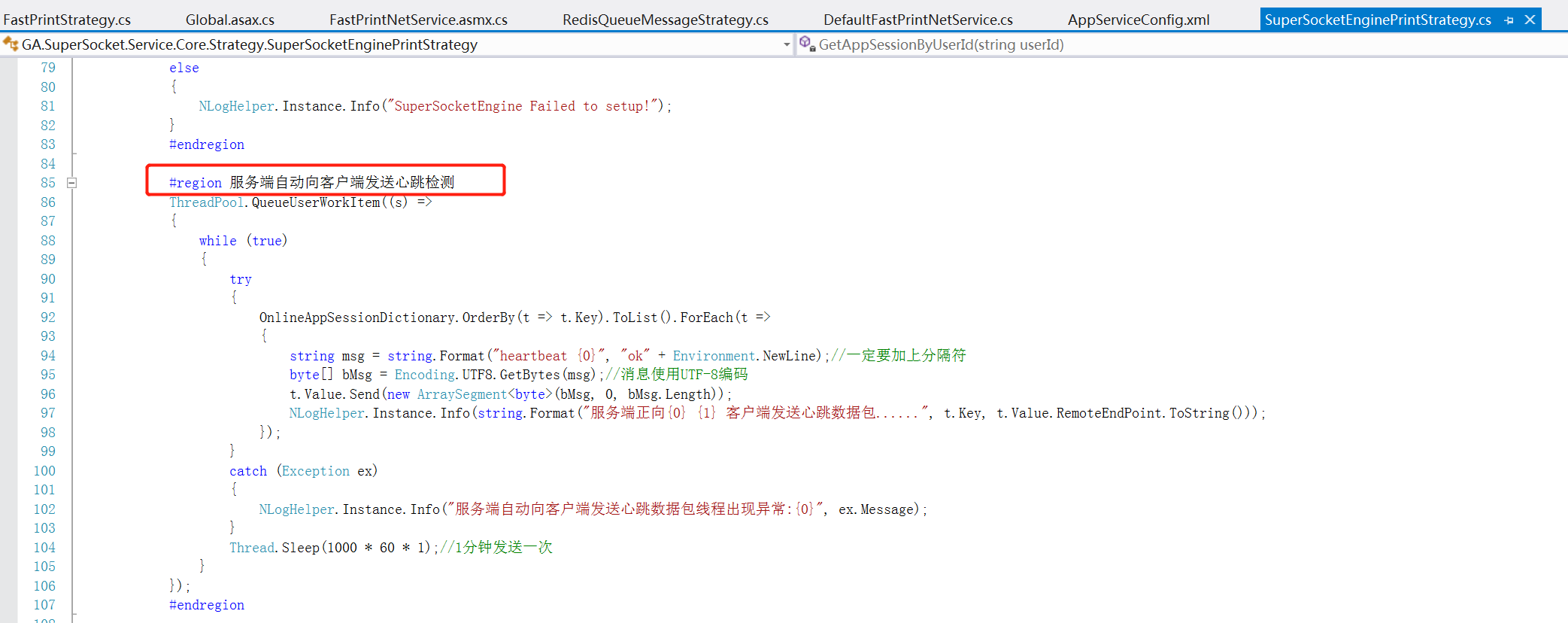
启动socket服务、监听会话连接、监听来自客户端的消息发送、监听会话关闭

会话连接： tcpServerEngine\_NewSessionConnected,当客户端一旦上线后socket服务会自动监听到连接请求并创建socket会话，即进入到tcpServerEngine\_NewSessionConnected方法里，即MyAppSession是客户端请求连接时自动创建的session数据

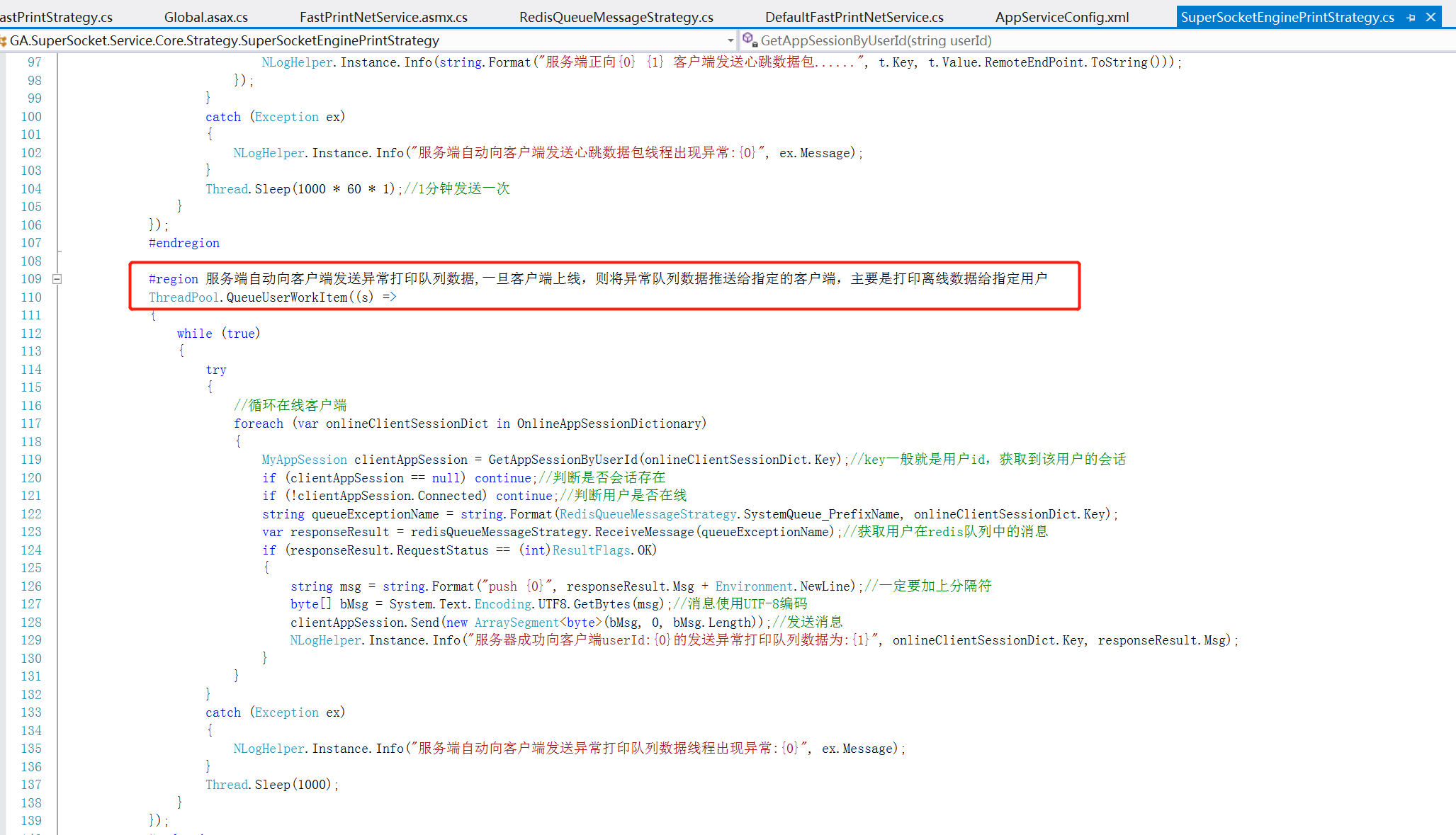


* **定时向在线的客户端发送心跳数据包**

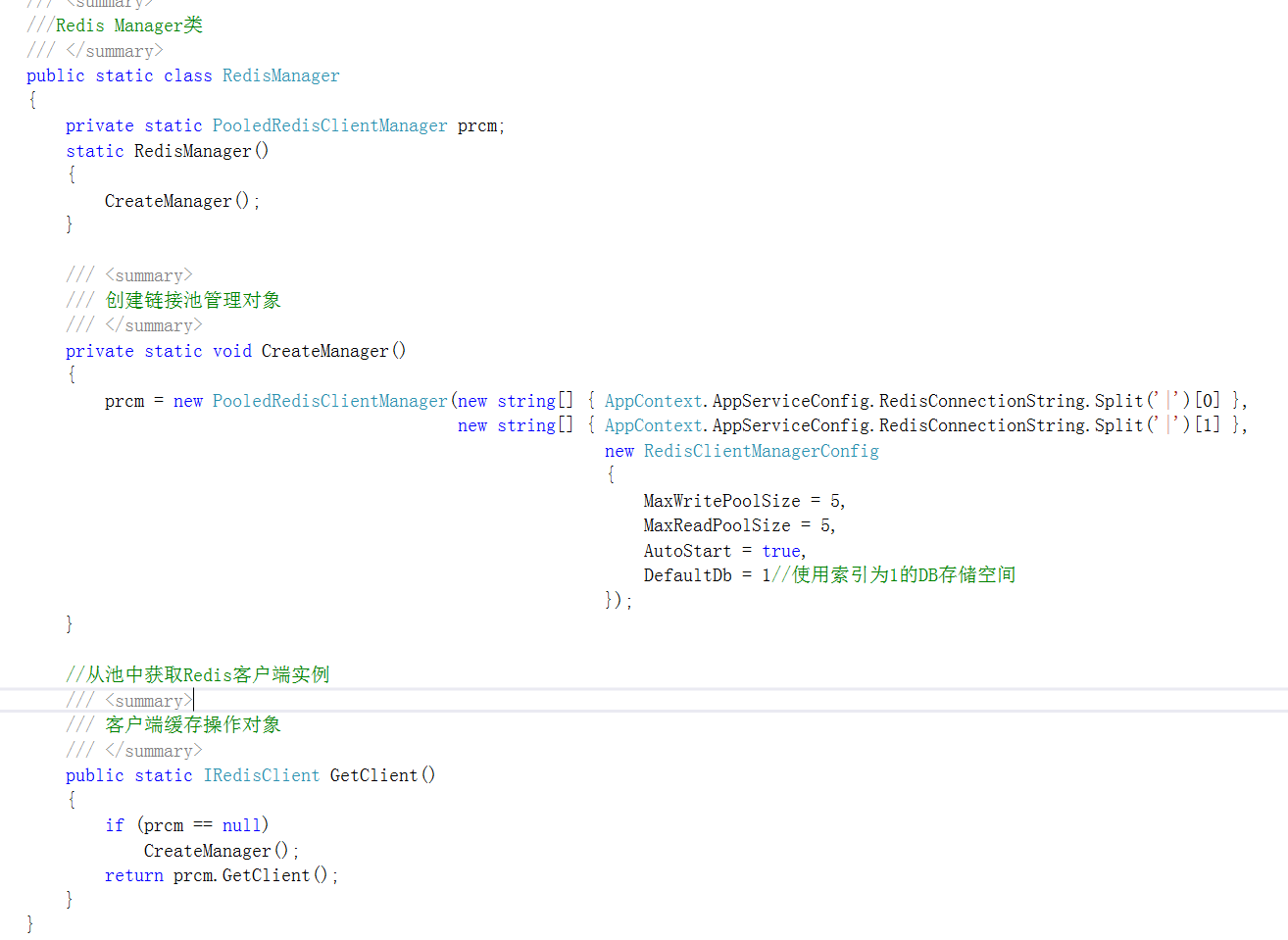
主要是为了维持与客户端的长连接，因为tcp连接会因为网络等原因出现断连情况，如果断连会导致服务端与客户端无法通信，就好比手机的电话卡处于无服务状态，手机信号其实也是不断地通过发送或接收数据维持电话信号的。具体请看流程图**3 socket通信流程图。**



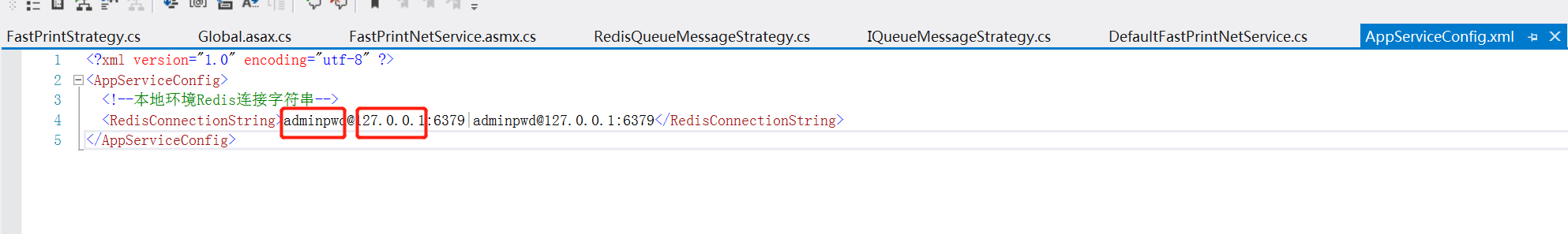
1. **检测登录用户上线后自动打印离线消息发送给指定用户**



1. 文件名：RedisQueueMessageStrategy.cs
2. **初始化redis对象**



注意： 初始化对象时会因两个问题进行报错，一个是配置文件要设置为ip,另外一个问题是redis部署未成功，比如服务未跑起来或者redis验证密码没设置。



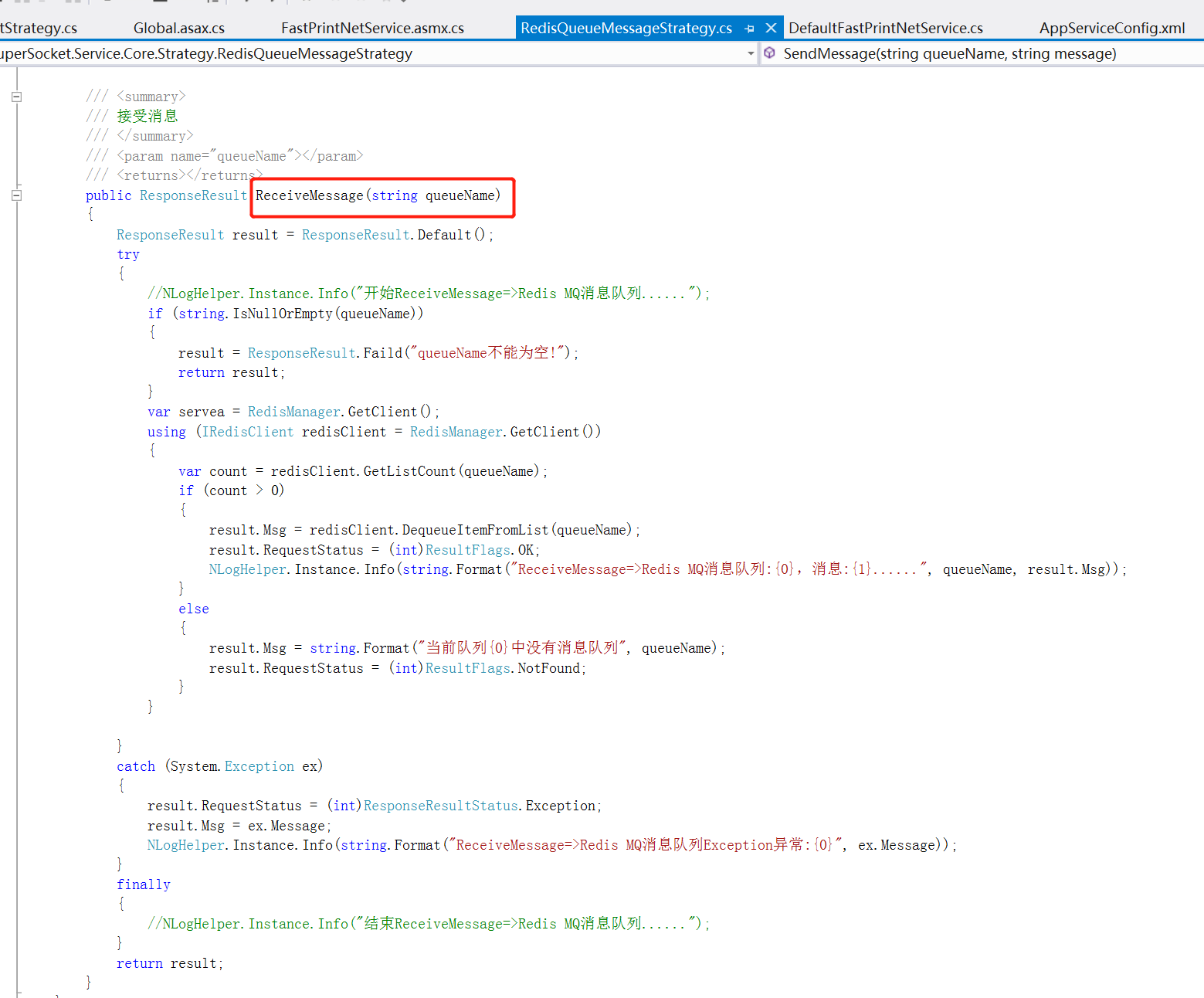
1. **存储离线消息**

配合SuperSocketEnginePrintStrategy文件中的程序会通过这里的SendMessage方法发送消息过来，用于存储离线消息到redis中



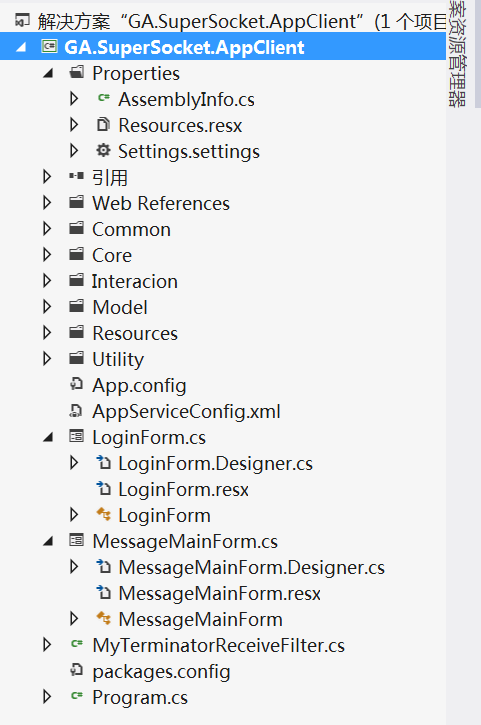
1. **取出离线消息**

配合SuperSocketEnginePrintStrategy文件中的程序检测到用户一旦上线后，会通过这里的ReceiveMessage方法从redis取出存储的离线消息

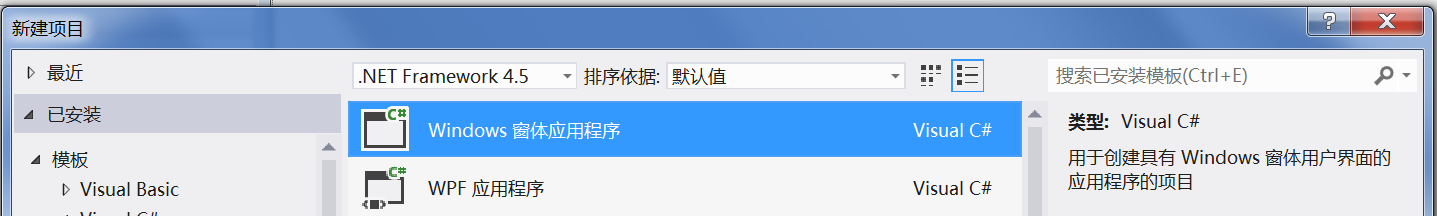


## PC端

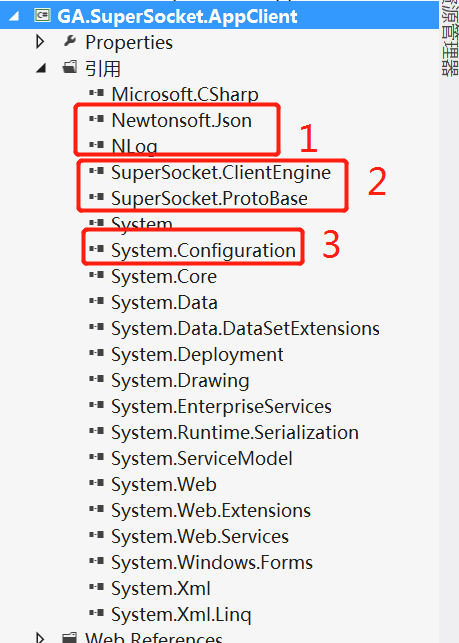
### 框架解读



1. **GA.SuperSocket.AppClient**属于Windows应用程序，.net framework 4.5版本



1. **引用**



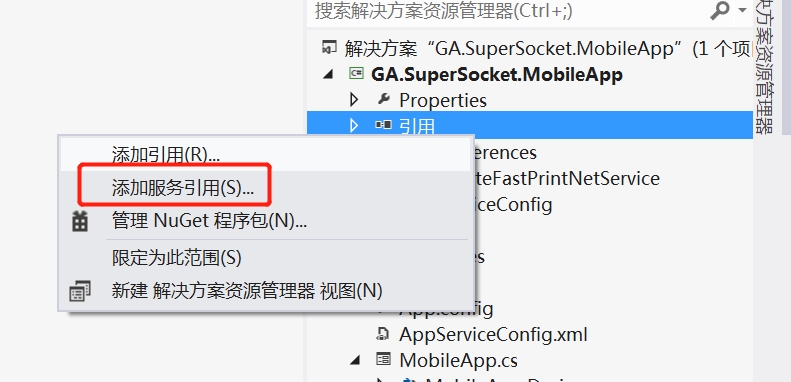
（1）序列化对象和日志组件库;

（2）rsocket核心组件库；

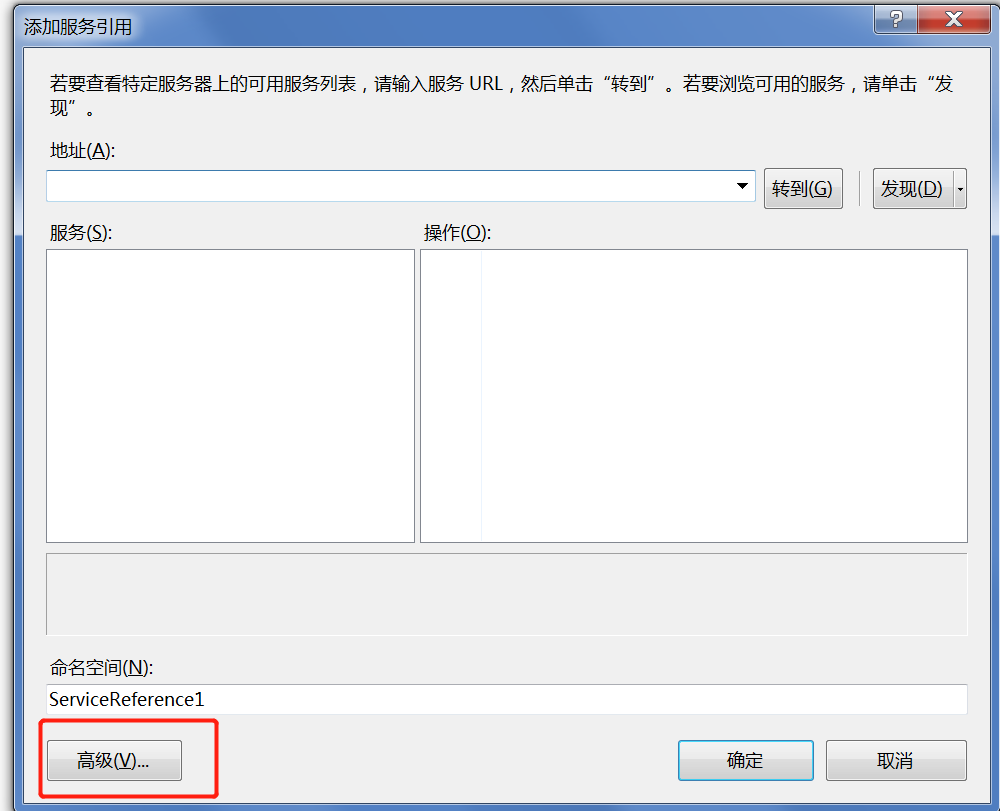
（3）读取App.config、Web.config的库

1. **Web References**

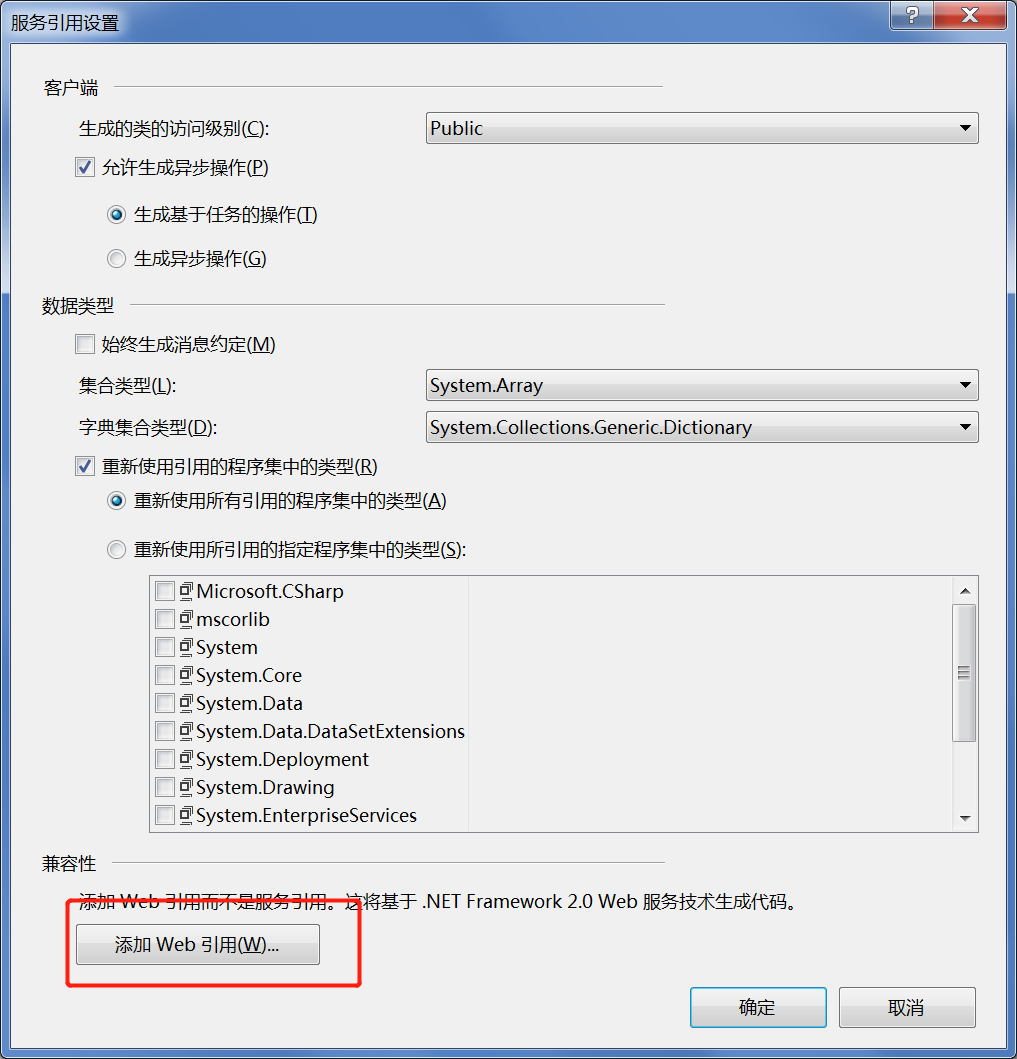
引用服务端部接口，采用Web引用，如下图所示：



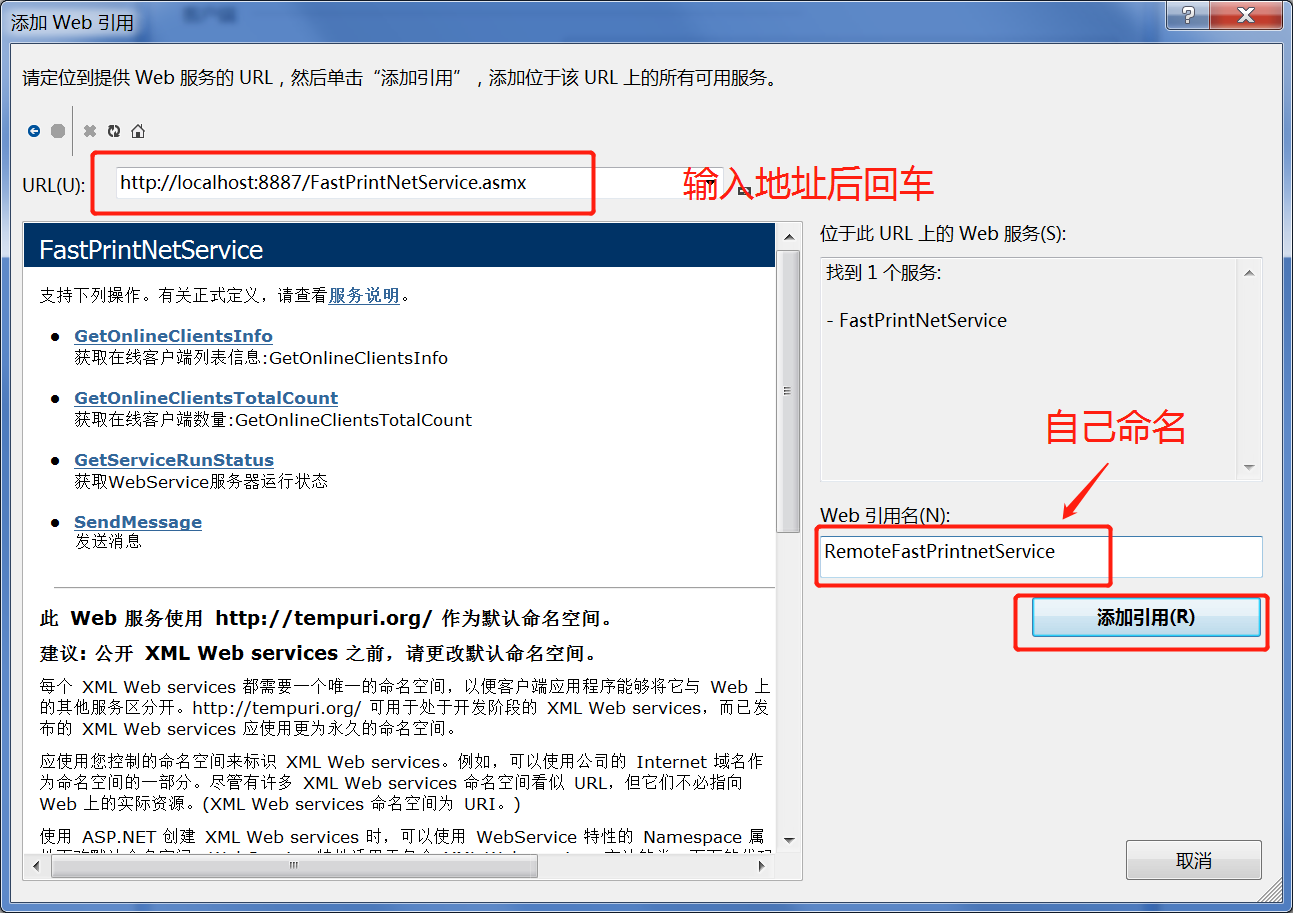
选择高级



选择Web引用



输入地址，比如<http://localhost:8887/FastPrintNetService.asmx>



1. **Core**

主要是用来加载配置文件AppServiceConfig.xml，获取到远程web服务IP地址和Socket服务器IP地址及端

1. **Model**

自定义的实体类

1. **Resources**

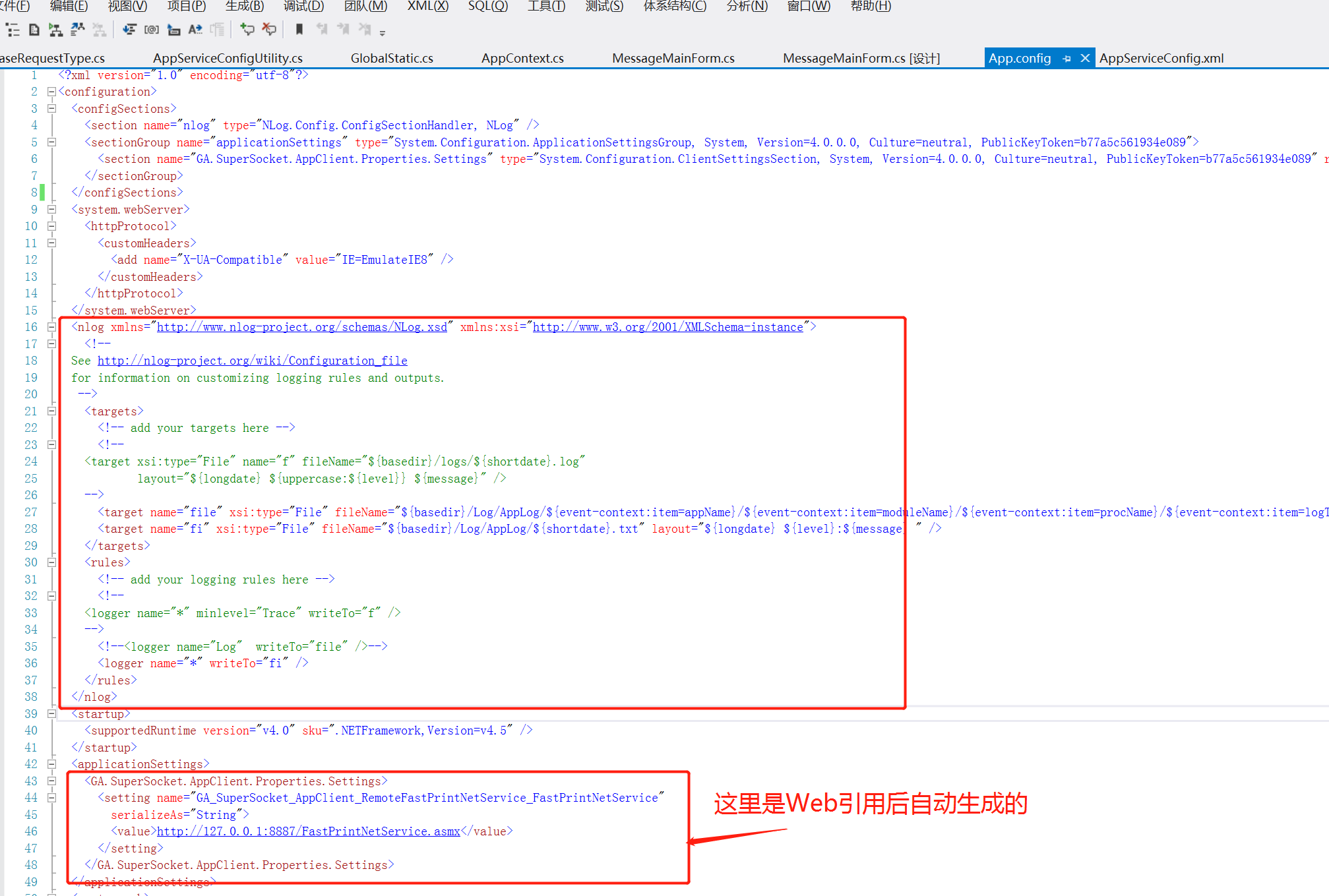
放置Winform客户端的一些图标资源

1. **Utility**

辅助及扩展类

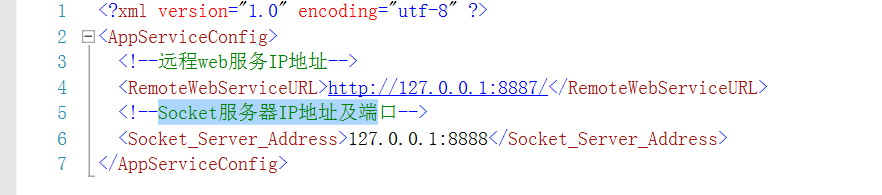
1. **App.config**

主要是配置日志信息（日志组件库、打印日志的输出格式）以及远程服务地址



1. **AppServiceConfig.xml**

xml配置文件，配置了远程Web服务IP地址以及Socket服务地址



1. **LoginForm.cs**

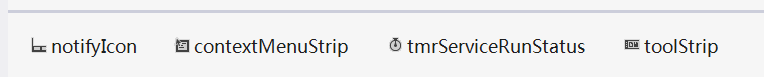


**登录窗口：**主要用于登录操作

1. **MessageMainForm.cs**



**消息跟踪窗口：**主要是展示客户端与客户端、服务端与客户端等之间的全部消息内容在这里展示。



**notifyIcon：**用于登录成功后进入到MessageMainform后在窗口顶部显示其登录账号；

**contextMenuStrip：**功能小菜单，窗口最小化可显示主界面、重启以及退出程序；

**tmrServiceRunStatus：**定时器，定时检查Web站点（比如：8887端口）服务的运行状态，若异常则重新初始化程序；

**toolStrip：**显示连接Socket服务（比如：8888端口）的信息以及状态。

1. **MyTerminatorReceiveFilter.cs**

将数据包的数据进行转化，转化为StringPackageInfo类的格式。注意：发送数据时协议上规定必须以“/r/n”进行结束。

1. **Packages.config**

引用的包文件，包含了各个引用的文件包名、版本、.net framework版本

1. **Program.cs**

程序启动类文件

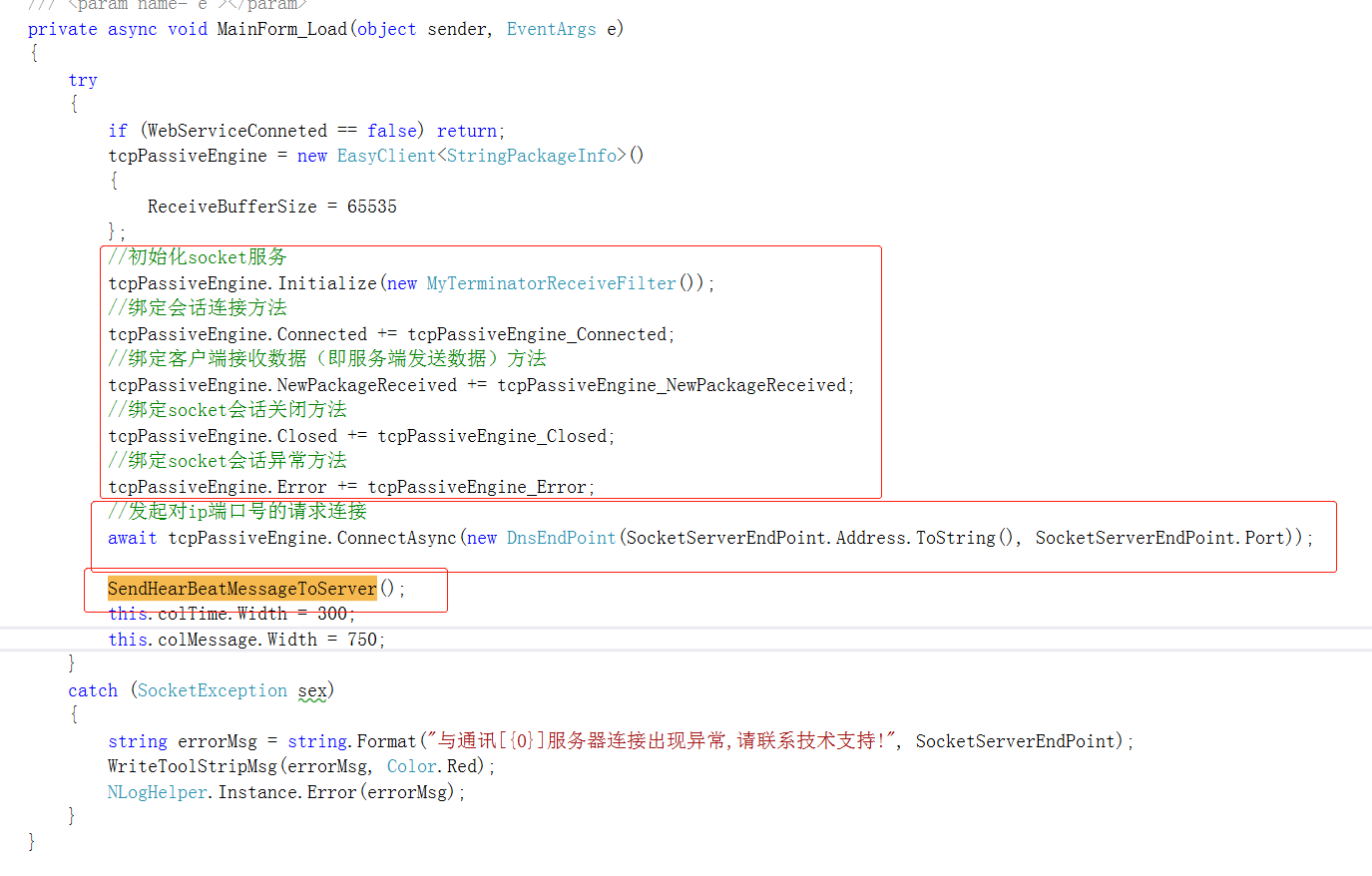
### 代码解读

1. 文件名：Program.cs是程序启动文件



程序启动后显示**登录窗口**，填写用户名、密码登录成功后进入到**消息窗口**。

1. 文件名：MessageMainForm

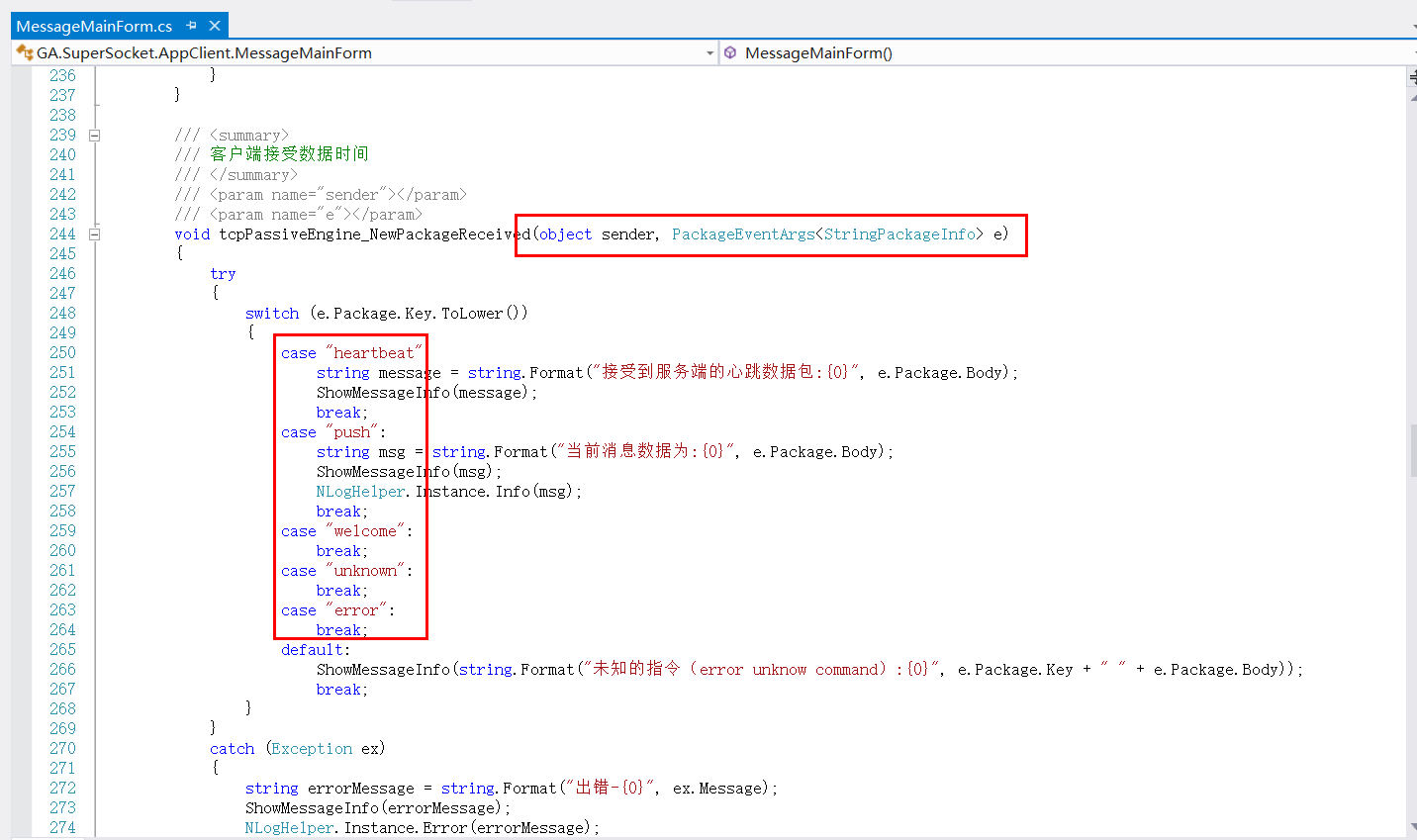


核心程序就是：

（1）初始化socket并绑定相关事件；

（2）发起对socket服务的连接请求；

（3）发送连接成功后的确认信息给服务端



注意：如果客户端接受数据异常，程序一直走default的话，那说明是有问题的。可查看红框中的参数是否有报错问题，一般这种情况是由于客户端和socket服务（例如：8888端口）的tcp连接不成功导致的。

## 移动端

GA.SuperSocket.MobileApp此项目比较简单，主要是给在线的指定用户发送消息。

# Redis部署

下载redis，下载地址：<https://redis.io/download>



下载安装Redis服务后，打开Redis服务安装的路径，如下图所示：



（1）**打开命令窗口**

本人的安装路径是C：\\Program Files\Redis下，Shift + 右键鼠标操作，点击“在此处打开命令窗口”，进入到命令窗口。

（2）**启动服务命令**

输入下图命令，如果出现下面的图示即表示启动成功。

命令：redis-server redis.windows.conf



（3）**安装服务**

由于上面虽然启动了redis服务命令，但是只要关闭cmd窗口，redis服务就会停止。所以要把redis设置成windows下的服务。

命令：redis-server --service-install redis.windows-service.conf --loglevel verbose



（4）启动服务

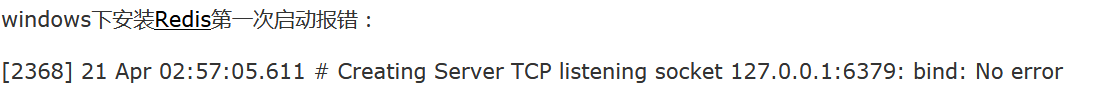
打开本机上的 **任务管理器**->**服务，**找到Redis服务并启动即可。

注意：启动类型要设置为 自动，否则每次电脑开机，此服务不会自动启动。



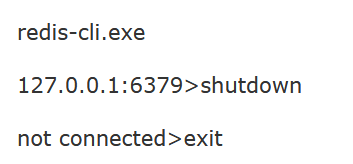
**Redis服务安装时的报错问题：**

如果出现下图报错，即表示该端口下的服务已被使用，则需要重新启动服务即可。



解决办法：

按照上面的第（1）操作的命令窗口下按顺序输入三次命令：redis-cli.exe、shutdown、exit,然后进入到上面的第（2）步操作



Redis操作命令学习地址：<https://www.runoob.com/redis/redis-security.html>

地址中包含了我们要给redis设置访问密码的命令行，如下图所示：



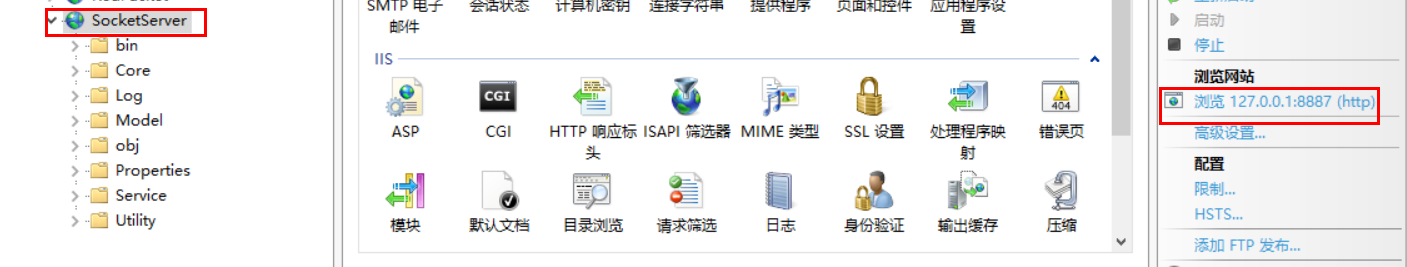
redis学习地址：<https://www.cnblogs.com/weifeng1463/p/9713594.html>

该地址包含了redis客户端的下载（[https://pan.baidu.com/s/1QvjG30IV-MJFPTF-oV9nVw](https://pan.baidu.com/s/1QvjG30IV-MJFPTF-oV9nVw" \t "_blank)

）以及客户端连接redis的操作方法

# 本地IIS发布流程

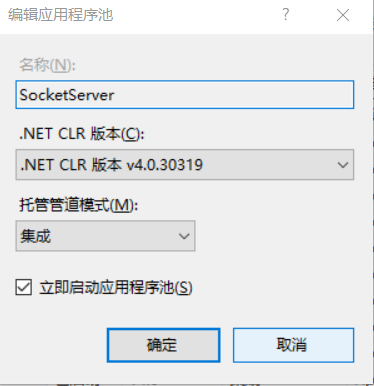
* 服务端发布



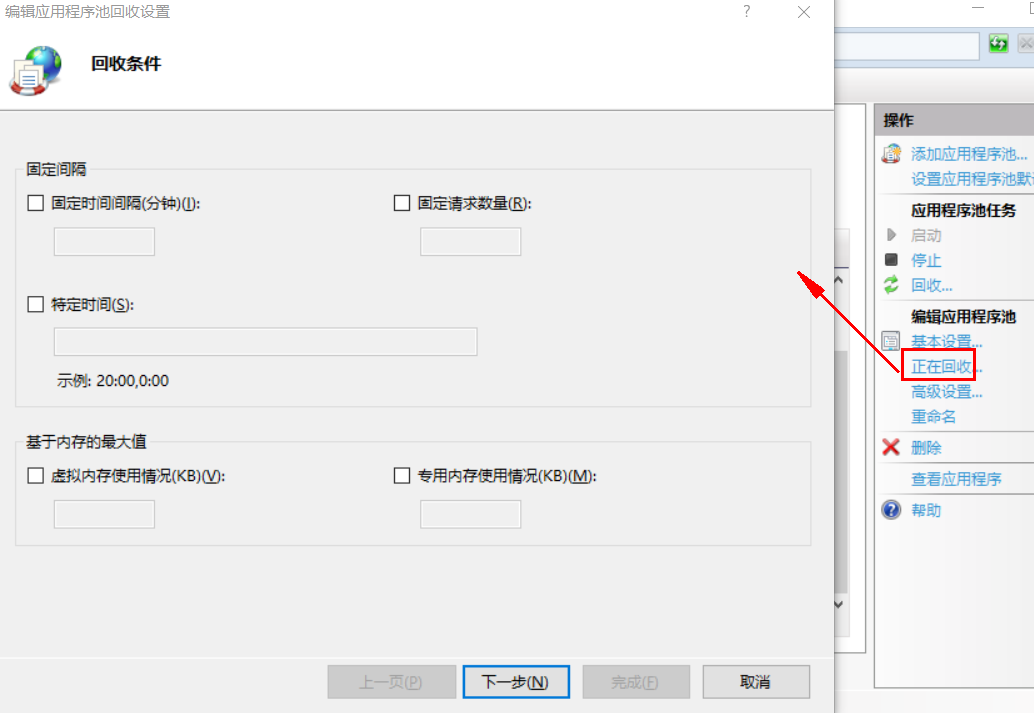
配置站点为：SocketServer，IP:127.0.01，端口号：8887

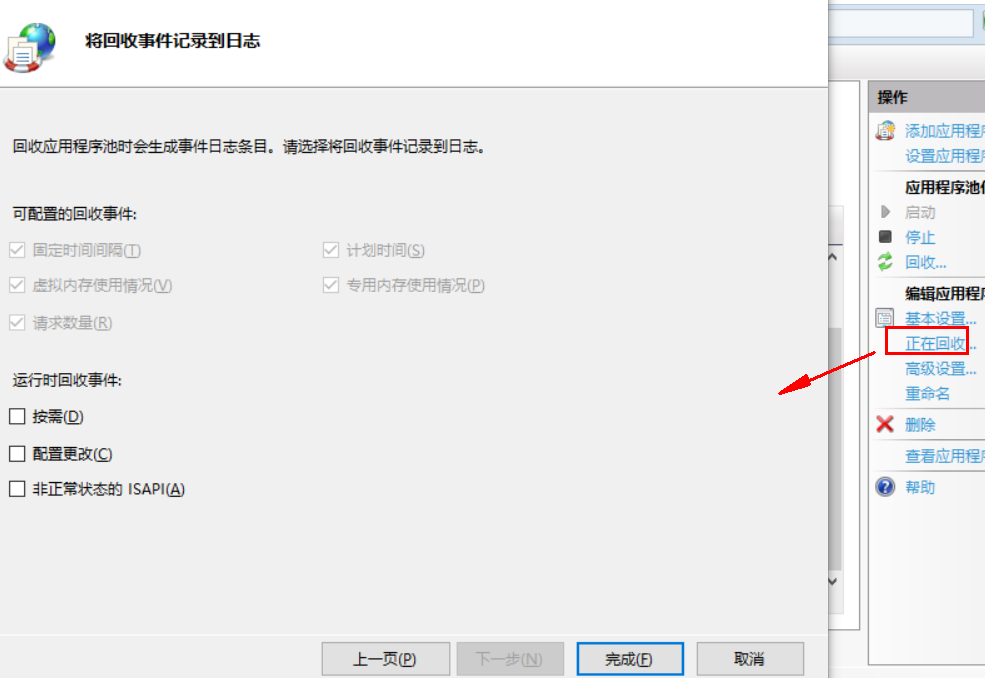
注意：ip要使用具体的IP号，不要使用localhost,不然可能会导致 客户端与服务端的Socket服务(8888端口)连接失败

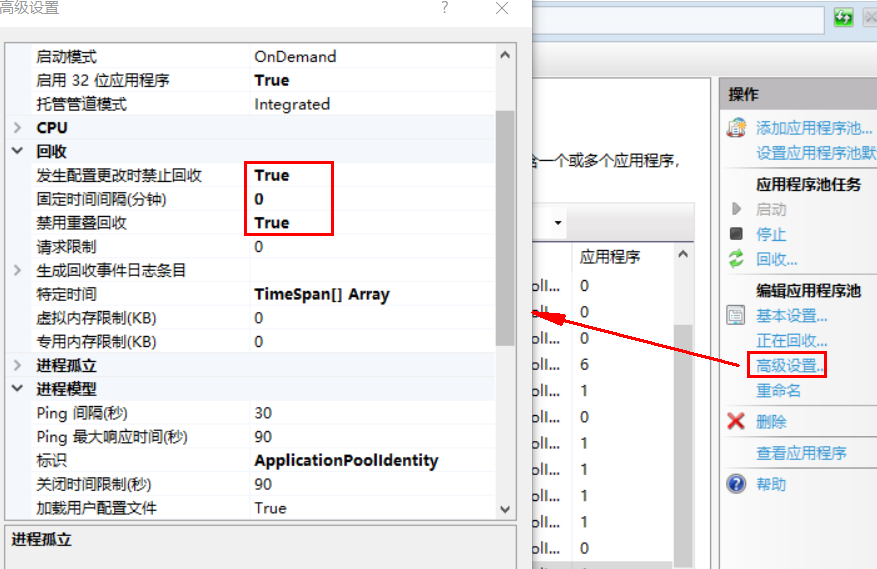
应用池配置



若我们配置的站点长时间未被访问会被IIS自动回收，防止IIS自动回收应用程序进程的基本设置。







* 客户端发布

可将**MobileApp**和**AppClient**项目的bin文件编译好后放在其他文件夹下，运行里面的**.exe**文件即可进行多个客户端运行。

