Web 服务器设计

# 软件目的

设计一个支持多线程的处理的Web服务服务端软件，目的在于：

1. 熟悉HTTP协议
2. 进一步学习软件设计方法
3. 学习通过网络端口实现进程间的通信

# 软件总体说明

设计一个支持多线程的处理的Web服务服务端软件。功能上，要求服务器参数可配（IP，端口，主目录），能响应HTTP请求（包括HTML文件和其他常用类型的文件），能显示请求和处理结果，并且具有一定的错误处理能力（例如当无法成功定位文件时，根据错误原因，做出相应的错误提示）。用户体验上，要求具有比较良好的用户体验（内部多线程，界面友好易操作）。

# 需求分析

* 1. 可配IP、端口和主目录
  2. 支持服务的开关
  3. 屏幕输出每个请求来源（IP、端口、HTTP请求命令行）
  4. 能根据HTTP请求发送相应的响应报文（HTML文件和其他常见类型的文件（如TXT和JPG等）），并在屏幕上输出相应的处理结果。
  5. 能根据错误请求的错误类型，进行相应的错误相应。
  6. 日志（请求和响应日志）
  7. 良好GUI

# 概要设计

## 设计说明

程序为实验性程序，主要目的为熟悉和掌握HTTP协议，在安全方面，没有要求。

### 设计理念

1. 模块独立

要求每一个单一的模块尽量减少对其他的模块和库的依赖性。如果一个模块能不依赖其他模块而单独的实现功能，就单独的实现。如果能不依赖其他的库来实现，尽量不依赖。

1. 良好的用户体验

要求服务器能比较好的响应用户的请求，对于错误请求，给出相应的错误代码。并且在服务器端，要求存储记录所有的请求。

### 平台选择

语言：C++

操作系统：Microsoft Windows 7, 64 bit

IDE: Microsoft Visual Studio 2010

## 模块分析设计

### 流程图

### 模块初步分析

根据系统的需求分析和流程图，可以观察到，系统主要分为四大模块。一个是界面模块，用以响应用户操作。二是服务器管理模块，负责Web服务的配置、管理当前的服务器连接和日志记录。三是链接管理模块，负责监听端口和管理链接。四是链接通信模块，对请求的负责单条TCP链接的请求和响应。如下：

1. 界面模块（MainWindow）

响应用户的操作和显示Web服务器的状态（当前服务开关状态、HTTP请求和响应结果），包括如下：

1. 开关服务
2. 配置服务器ip、端口和主目录
3. 查看日志
4. 显示状态信息
5. 服务器管理模块（WebMgr）

负责管理服务器，包括：

1. 加载、修改和保存Web服务器配置（IP、端口、主目录）
2. 开关服务器
3. 日志记录
4. 链接管理模块（cntMgr）

负责监听端口、管理链接和具体的开关服务

1. 链接响应模块（HttpResponser）

负责单条TCP链接的请求和响应。

## 数据结构

### 配置和日志存储结构

由于配置和日志都不是很大的文件，采用文本文件存储即可。

（1） 配置文件

配置文件存储服务器的参数配置，包括服务器的IP、端口和主目录，每行存一个参数即可。如：

222.20.105.112

80

E:\

（2） 日志文件

日志文件用来存储Web服务器的请求和响应信息，记录每一条请求的时间、客户机IP地址、端口号、请求命令行和处理响应结果。每条记录包括多行，第一行是请求的时间、客户机IP、客户机端口号和处理响应结果，接下来的几行记录的就是客户机的请求报文，然后隔一个空行之后再记录下一条记录。

### 内存数据结构

内存中需要处理的数据主要有请求报文数据和服务器参数数据。服务器参数数据用常规的处理方法即可。对于请求报文，有一个排队问题。虽然对于每个请求采用的是多线程来处理的，但是多线程也是有限的，当请求过多时，为了不使服务器崩溃，需要采用一定的措施来处理请求过多，服务器无法负载的情况。这里我采用经典的排队方法：

假设Web服务器最多能同时处理的请求数（即线程数）为MAX\_ON\_PROCESSING，当请求超过这个数时，采用队列的方法来存储新来的请求，队列长度为MAX\_QUEUE，当超过这个数时，丢弃请求。出于方便，直接采用C++的容器（Vector）来实现队列的操作。

# 详细设计

### 模块关系图

### 界面模块（MainWindow）

界面模块通过调用服务管理模块实现响应用户的操作，通过服务管理模块发射给界面模块的信号来更新实现信息。

该模块是程序的顶层模块，不需要提供供其他模块使用的接口，但是由于需要来自服务管理模块来的更新信息（HTTP请求和响应结果），需要提供一个槽来接收这类信息。

槽：

更新信息：void On\_NewRequest\_Processed(string result);

### 服务管理模块（WebMgr）

服务管理模块向上（界面模块）需要提供如下服务：

* + 1. 服务的开关
    2. 获取和设置服务器的设置
    3. 获取日志目录

同时，当服务管理模块处理完一条请求的时候，需要向界面模块发送相应的信号。

由于当解析模块完成了一项解析时，需要得到解析结果，故需要一个来接收来自解析模块处理信息的槽。

另外， 该模块还需要有如下功能：

1. 加载、修改和保存Web服务器配置（IP、端口、主目录）
2. 为每个请求新建解析和响应线程（该线程里完成解析、生成响应报文和发送响应报文）
3. 日志记录

总而言之，该模块需要：

槽：接收来自解析模块的处理结果

void On\_Process\_Finished(int code, vector<Byte>)

接口函数：

1. 开启服务：bool Start()
2. 关闭服务：bool Stop()
3. 获取服务器配置设置：void GetServerSet(string &ip, unsigned int &port, string &webPath)
4. 获取日志目录：string GetLogPath()

信号：发送处理结果信号

void newRequestResult(string result)

### 链接管理模块（CntMgr）

负责监听端口和管理链接资源。

其中主要涉及到如下3个问题：

1. 当服务器过载时，链接排队问题

2. 当一个链接响应完成时，线程资源释放和链接资源释放问题

第一个问题，直接控制QTcpServer中的maxPendingConnections来控制最大的链接排队数，超过这个数时，QTcpServer会自动丢弃该请求。

第二个问题涉及到如何管理各个链接的问题。这里，我们采用表的方式来管理。假设能够同时处理的最大链接数为MAX\_SIM\_PROCESSING，则给每个链接响应器（HttpResponser）从小到大分配一个唯一的id，范围为0~MAX\_SIM\_PROCESSING-1。当HttpResponser发来关闭链接信号时，搜索所有的id，释放HttpResponser所占用的资源。

### 响应 器（HttpResponser）

解析直接用Qt的线程类QThread实现。为了该类的构造函数需要传递2个参数（链接和主目录），然后使用Qthread的start函数来启动线程。完成解析后，发送一个包含处理结果的信号。

1. 负责监听链接，当断开链接时，给CntMgr发送链接断开信号，请求释放资源。
2. 接收和发送报文。
3. 解析报文（新线程）

# 软件实现

# 测试

# 参考文献