**系统设计文档**

**版本<1.0>**

**时间<2016.12.21>**

**目录**

**一、 系统需求分析** 1

**1.1功能需求分析** 1

**1.2性能需求分析** 1

**1.3运行环境需求分析** 2

**二、 系统设计** 3

**2.1系统整体框架** 3

**2.2服务器端设计** 4

**2.2.1服务器端概述** 4

**2.2.2服务器端功能** 4

**2.3客户端设计** 5

**2.3.1客户端概述** 5

**2.3.2功能模块** 5

**2.4数据结构设计** 6

**2.4.1宏定义设计** 6

**2.4.2协议数据结构设计** 7

**2.4.3实体数据结构设计** 9

**三、 系统实现** 10

**3.1服务器端实现** 10

**3.1.1 网络I/O模型** 10

**3.1.2 音乐列表的查询获取** 10

**3.1.3 音乐播放/停止** 10

**3.2客户端实现** 10

**3.2.1 Socket操作线程** 10

**3.2.2 数据转换** 11

**四、 运行界面展示** 13

1. **系统需求分析**

需求分析是系统定义阶段的最后一步，确定系统必须完成哪些工作，也就是在开发前，以目标用户的需求为依据，对客户端分不同层次进行递进和整理，最后对目标系统提出完整、准确、清晰、具体的要求。

本系统的主要目的是，在局域网下，为用户提供远程获取电脑上的音乐列表，以及可对音乐进行播放、停止的服务。

**1.1功能需求分析**

本文主要是针对Android用户进行的系统设计，初期阶段的主要工作有：

1. 登录功能：对使用本软件的用户进行身份验证。
2. 获取音乐列表功能：客户端发送获取音乐列表的请求，服务器通过该请求后，返回音乐列表的数据。
3. 播放、停止音乐功能：用户通过软件向服务器发送停止/播放某一首音乐的请求，服务器对该请求进行处理后，返回音乐停止/播放的状态。
4. 退出功能：关闭客户端，在服务器的用户列表中清除该登录用户。

**1.2性能需求分析**

1. 可用性需求：考虑到系统的可用性，对用户来说操作起来应简单易学，对系统自身来说，故障检测、恢复时间等都应该很小。
2. 性能需求：本系统采用的是C/S模式，为保证系统的运行速度，本地单个操作运行时间要少于100ms，其中服务器端的响应时间控制在30ms左右。
3. 可移植性和可修改性需求：系统在设计上要考虑到后期功能拓展的需要，支持二次开发，方便迭代。
4. 一致性需求：保持客户端和服务器端的数据一致，让系统稳定运行。

**1.3运行环境需求分析**

1. 客户端运行环境：考虑到客户端应满足在大多数Android手机上运行的情况，客户端所支持的Android版本在4.1.2-7.1.1之间。
2. 服务器端运行环境：支持在Windows环境下运行。
3. **系统设计**

**2.1系统整体框架**

本系统采用C/S模式，包含客户端和服务器端，服务器端主要为用户提供各种访问服务，而客户端主要负责与用户交互，并通过TCP连接与服务器端进行信息交互。

系统框架如下图：

系 统

管理员

用户

请求

服务器端

Android

客户端

响应

**2.2服务器端设计**

**2.2.1服务器端概述**

服务器端存储了本系统运行过程中的所有数据，管理员可以对数据进行增加、删除、查询、修改操作，同时服务器端会处理客户端的请求，并处理、反馈数据。

**2.2.2服务器端功能**

服务器端负责处理客户端的请求，主要功能包括：登录验证、查询音乐列表、播放/停止音乐、退出。

1. **登录验证功能**

获取用户的登录请求，并在用户文件中对比请求中的用户名和密码，如果匹配，返回给客户端登录成功，否则返回登录失败。

1. **查询音乐列表功能**

获取用户的请求，对音乐文件夹中的所有音乐进行遍历，并将每首音乐的名字返回给客户端。

1. **播放/停止音乐功能**

对获取到的用户请求进行处理，如果是播放请求，则对这首音乐进行播放，如果是停止请求，则对这首音乐进行停止。

1. **退出功能**

在已登录用户列表中，清除该登录用户的Session。

**2.3客户端设计**

**2.3.1客户端概述**

客户端负责与用户进行交互，对用户的操作进行处理，封装相关请求，通过网络完成与服务器端的信息交互，并对返回的数据进行处理。

**2.3.2功能模块**

客户端包括三个模块：登录模块、音乐模块、退出模块。

**1.登录模块**

用户输入用户名和密码，点击登录，客户端会把这个请求进行封装，并且发送给服务器端，之后接收到返回的数据，弹出对话框，提示用户是否登录成功。

**2.音乐模块**

此模块可分为两个部分。第一，用户点击获取音乐列表，会向服务器端发送一个请求，之后接收到返回的数据，把这些数据进行解析后，逐条添加到音乐列表中，并在界面进行显示；第二，用户点击一首音乐的播放/停止按钮，客户端会对这首音乐是否可播放/停止进行判断，如果可以，则发送一个请求给服务器端，然后对返回的数据进行处理，弹出对话框，提示用户是否播放/停止成功。

1. **退出模块**

用户点击退出，关闭客户端，并且发送一个退出请求到服务器端。

**2.4数据结构设计**

**2.4.1宏定义设计**

|  |  |  |
| --- | --- | --- |
| **宏名称** | **值** | **描述** |
| PKT\_LOGIN | 0 | 登录包 |
| PKT\_LOGOUT | 1 | 退出包 |
| PKT\_PLAY\_START\_STOP | 2 | 播放/停止包 |
| PKT\_PLAY\_START\_STOP\_REPLY | 3 | 播放/停止回复包 |
| PKT\_LOGIN\_REPLY | 4 | 登录回复包 |
| PKT\_GET\_MUSIC\_LIST | 5 | 获取音乐列表包 |
| PKT\_GET\_MUSIC\_LIST\_REPLY | 6 | 获取音乐列表回复包 |
| LOGIN\_FAIL | 0 | 登录失败 |
| LOGIN\_SUCCESS | 1 | 登录成功 |
| PLAY\_START | 0 | 播放 |
| PLAY\_STOP | 1 | 停止 |
| PLAY\_START\_FAIL | 0 | 播放失败 |
| PLAY\_START\_SUCCESS | 1 | 播放成功 |
| PLAY\_STOP\_FAIL | 2 | 停止成功 |
| PLAY\_STOP\_SUCCESS | 3 | 停止失败 |
| UNLOGIN\_STATE | 0 | 未登录状态 |
| LOGIN\_STATE | 1 | 登录状态 |

**2.4.2协议数据结构设计**

**包头(PktHeader)**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **类型** | **长度** | **描述** |
| pktLen | int | 1 | 包长 |
| pktType | int | 1 | 包类型 |

**登录包(LoginPkt)**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **类型** | **长度** | **描述** |
| header | PktHeader | 1 | 包头 |
| name | char | 20 | 用户名 |
| password | char | 20 | 密码 |

**退出包(LogoutPkt)**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **类型** | **长度** | **描述** |
| header | PktHeader | 1 | 包头 |
| name | char | 20 | 用户名 |

**登录回复包(LoginReplyPkt)**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **类型** | **长度** | **描述** |
| header | PktHeader | 1 | 包头 |
| ret | int | 1 | 是否登录成功 |

**播放/停止包(PlayStartStopPkt)**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **类型** | **长度** | **描述** |
| header | PktHeader | 1 | 包头 |
| type | int | 1 | 播放/停止类型 |
| musicName | char | 30 | 音乐名 |

**播放/停止回复包(PlayStartStopReplyPkt)**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **类型** | **长度** | **描述** |
| header | PktHeader | 1 | 包头 |
| ret | int | 1 | 播放/停止是否成功/失败 |

**获取音乐列表包(GetMusicListPkt)**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **类型** | **长度** | **描述** |
| header | PktHeader | 1 | 包头 |

**获取音乐列表回复包(GetMusicListReplyPkt)**

|  |  |  |  |
| --- | --- | --- | --- |
| **字段** | **类型** | **长度** | **描述** |
| header | PktHeader | 1 | 包头 |
| musicName | char | 30 | 音乐名 |

**2.4.3实体数据结构设计**

**用户实体(User)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **是否为空** | **默认值** | **描述** |
| name | char | 否 | 无 | 用户名 |
| password | char | 否 | 无 | 密码 |

**Session实体(Session)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **是否为空** | **默认值** | **描述** |
| name | char | 否 | 无 | 用户名 |
| connection | Socket | 否 | 无 | 套接字 |
| state | int | 否 | 0 | 登录状态 |
| payload | char | 是 | 无 | 消息缓冲区 |
| pktLen | int | 是 | 0 | 实际包长 |
| recvPktLenSize | int | 是 | 0 | 已接收的长度 |
| recvPayloadSize | int | 是 | 0 | 消息缓冲区已接收长度 |

**音乐实体**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **是否为空** | **默认值** | **描述** |
| musicName | String | 否 | 无 | 音乐名 |

1. **系统实现**

本系统是基于Socket编程实现的，服务器端使用C语言编写，客户端使用Java语言编写，通过TCP连接来发送、接收数据，数据统一转换为byte数组进行相互传递。

**3.1服务器端实现**

**3.1.1 网络I/O模型**

网络I/O模型采用select模型，实现多路复用，对每一个套接字进行监听，如果有响应，即可进行处理，提高服务器的性能，并且在Session结构体中通过加入缓冲区，在包头中加入包长变量，实现对数据的组包操作，以及非阻塞式接收数据。

**3.1.2 音乐列表的查询获取**

通过调用GetCurrentDirectory、SetCurrentDirectory、FindNextFile和FindFirstFile函数，遍历指定音乐文件夹下的文件，获取对应的音乐文件名，返回给客户端。

**3.1.3 音乐播放/停止**

为mciSendString函数传入播放、停止对应的命令参数，实现对音乐文件的播放或者停止操作，然后获取到mciSendString函数返回的结果，如果是0，则返回给客户端播放或者停止成功，否则失败。

**3.2客户端实现**

**3.2.1 Socket操作线程**

实现一个自定义的Socket操作线程，主要完成四个工作。

1. **Socket连接**

为connect函数传入IP和端口参数，连接到服务器，并实例化数据输入流(DataInputStream)、输出流(DataOutputStream)对象。

1. **发送数据**

通过调用数据输出流的write函数发送指定长度的数据，并通过Handler对象返回给用户发送的状态。

1. **接收数据**

通过调用数据输入流的read函数先读取数据包长，再根据包长读取指定长度的数据，然后把两个数据进行拼接，并通过Handler对象返回给客户端接受到的数据。

1. **关闭网络操作**

关闭数据接收线程，释放数据输入流(DataInputStream)、输出流(DataOutputStream)，关闭Socket。

**3.2.2 数据转换**

为保持通讯时的数据一致，需要封装一个数据转换工具，主要有四个功能。

1. **int转byte数组**

将int转为低字节在前，高字节在后的byte数组。

1. **byte数组转int**

对int转byte数组的方法进行逆向处理即可。

1. **String转byte数组**

直接调用String的getBytes函数进行转换。

1. **byte数组转String**

通过String的构造方法String(byte[], “GBK”)进行转换。

1. **运行界面展示**

****

**登录界面**

App的logo与主题呼应。

界面清爽、简单，重点突出。易引起用户的使用兴趣。

只需输入用户名、密码，点击登录即可。

****

**音乐界面**

整个音乐列表排版整齐，友好的播放、停止状态显示框，使用方便、易懂。

****

**退出界面**

隐藏在右上角的设置菜单中，只需点击一下，即可退出当前应用。