

仿 Airprod

1. 实现功能

1.1 文件上传功能：

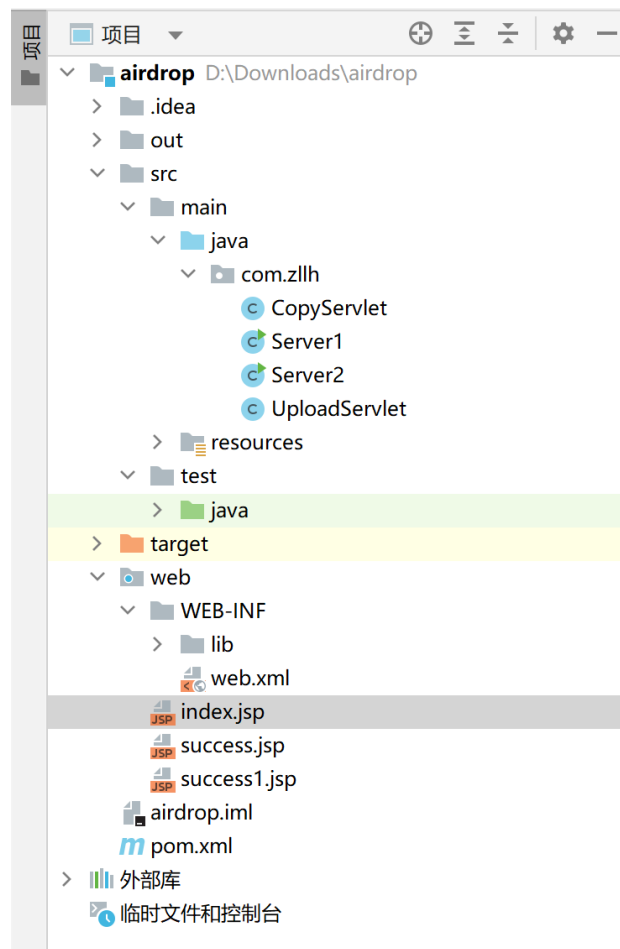
当客户端（如手机）与服务器在同一局域网下时，客户端可上传文件，格式为 txt 或者 jpg 和 png 的文档、图片。

1.2 剪切板功能：

当客户端与服务器在同一局域网下时，客户端在对应文本框中输入文本内容，即可在服务器的相应文件夹下生成文本文件，文件内容即客户端的剪切板内容。

2. 项目结构

项目结构如下：



其中运行 Server1.java 和 Server2.java 时，分别负责功能 1.1 和 1.2 的服务器启动，Server1 负责接收用户上传的文件，并写入到服务器相应位置；Server2 负责接收客户端的剪切板内容，并把剪切板的内容写到服务器相应位置。

当启动 Tomcat 后，进入 index.jsp 对应页面。客户端选择好上传内容并确认上传后，交由 UploadServlet 处理，结束后显示上传成功，根据用户是否继续上传的选择跳转至 index.jsp 或 success.jsp。

剪切板功能，在 index 页面，客户端在对应文本框中输入文本内容，复制请求交由 CopyServlet 处理，完成后显示成功提示。

3. 核心代码

3.1 至 3.4 为功能 1.1 核心代码，3.5.至 3.6 为功能 1.2 核心代码。

3.1 ip 和端口获取

```
DiskFileItemFactory factory = new DiskFileItemFactory();
    获取ServletFileUpload对象
ServletFileUpload upload = new ServletFileUpload(factory);

upload.setHeaderEncoding("UTF-8");

try {
    List<FileItem> fileItems = upload.parseRequest(req);
    for(FileItem item:fileItems){
        if (item.getFieldName().equals("upload"))
        {
            state=item.getString();
            System.out.println("从前端获取的内容是"+state);
        }
    }
    for (FileItem item : fileItems) {
        if (item.isFormField()) {
            if (item.getFieldName().equals("ip")) {
                url = item.getString();
                System.out.println("url是" + url);
            } else if (item.getFieldName().equals("port")) {
                port = Integer.parseInt(item.getString());
                System.out.println("端口号是" + port);
            }
        }
    }
}
```

3.2 获取要上传的文件的输出流

```
//
    获得上传的文件名
    System.out.print("当前的状态是:");
    System.out.println(state);
    uploadFileName = item.getName();
    long size = item.getSize();
    System.out.println("上传的文件名是" + uploadFileName);
    System.out.println("上传的文件大小是:" + size);
    message = uploadFileName + "#" + size + "#";
    open = item.getInputStream();
    socket = new Socket(url, port); // 创建socket
    out = (FileOutputStream) socket.getOutputStream();
    out.write(message.getBytes());
```

3.3 根据文件名，是 txt 还是图片格式，选择相应的流。

```
// 根据传输文件的类型选择适当的流
if(uploadFileName.contains("txt")){
    start = System.currentTimeMillis();
    InputStreamReader isr = new InputStreamReader(open, charsetName: "UTF-8");
    int n = isr.read(txt);
    System.out.println("要传输的东西是:" + new String(txt, offset: 0, n));
    out.write(new String(txt, offset: 0, n).getBytes());
    while(n != -1){
        n = isr.read(txt);
        if(n != -1) {
            out.write(new String(txt, offset: 0, n).getBytes());
        }
    }
    end = System.currentTimeMillis();
}

else {
    start = System.currentTimeMillis();

    int n = open.read(bytes, off: 0, bytes.length);
    System.out.println("要传输的东西是:" + new String(bytes, offset: 0, n));
    out.write(bytes, off: 0, n);
    while (n != -1) {
        System.out.println("看来文件比较大");
        n = open.read(bytes, off: 0, bytes.length);
        if (n != -1) {
            out.write(bytes, off: 0, n);
        }
        out.flush();
    }
    end = System.currentTimeMillis();
}
```

3.4 根据传输文件大小，临界值为 65535Byte，放至不同的目录下，防止上传的文件因重名被覆盖，接收的文件文件名为一串随机字符。

```

支持传输txt,png,jpg三种格式
String format = new String(nameBytes, offset: 0,n);
System.out.println(format);
String[] info = format.split( regex: "#");
System.out.println("传输过来的长度是"+info.length);
for (String s : info) {
    System.out.println(s);
}
if(info[0].contains("txt")){
    if(Long.parseLong(info[1])>65536){
        dir = "temp\\";
    }
    target = new File( pathname: path+dir+UUID.randomUUID().toString()+".txt");
    save = new FileOutputStream(target);
    while(n!=-1){
        n = in.read(bytes);
        if(n!=-1) {
            save.write(bytes, off: 0, n);
        }
    }
    System.out.println("接受成功!");
}
else if(info[0].contains("png")){
    if(Long.parseLong(info[1])>65536){
        dir = "temp\\";
    }
}

```

3.5 ip 和端口号获取，为提高剪切板的便利性，用户无需输入 ip 和端口号

```

FileOutputStream out = null;
// 通过类加载器获取相应的资源
InputStream is =CopyServlet.class.getClassLoader().getResourceAsStream( name: "config.properties");
Properties properties = new Properties();
properties.load(is);
String ip = properties.getProperty("ip");
int port = Integer.parseInt(properties.getProperty("port"));
Socket socket =new Socket(ip,port);

```

3.6 服务器响应阻塞等待客户端连接，将剪切板内容写入文本文档中。

```

// 处理客户端的请求
try {
    path = "D:\\Desktop\\Clipper\\";
    server = new ServerSocket( port: 3090);    // 服务器端口
    // 等待客户端的呼叫
    System.out.println("正在等待客户端的呼叫.....");


    while(true) {
        String name = UUID.randomUUID().toString();
        target = new File( pathname: path+name+".txt");
        save = new FileOutputStream(target);

        socket = server.accept();    // 阻塞
        System.out.println("接收到了客户端的请求");
        in = (FileInputStream) socket.getInputStream();
        // 接收数据
        byte[] b = new byte[64];
        int n = in.read(b);
        int start = (int) System.currentTimeMillis();
        while (n != -1) {
            save.write(b, off: 0, n);    // 写入指定地方
            n = in.read(b);
        }
        int end = (int) System.currentTimeMillis();
        System.out.println("接收成功,耗时: " + (end - start) + "毫秒");
    }
}

```

4. 单元测试

项目进行了 IO 测试和多线程测试。



```
CopyServlet.java × Server2.java × IOTest.java × StringTest.java × Test.java ×

import java.io.*;

public class IOTest {

    public static void main(String[] args) throws IOException {

        File f = new File( pathname: "D:\\Desktop\\1.txt");
        byte[] bytes = new byte[1024];
        //      FileOutputStream os = new FileOutputStream(f);
        //      String s = 1+"";
        FileInputStream is = new FileInputStream(f);
        is.read(bytes);
        //      FileReader fr = new FileReader(f);
        //      InputStreamReader ir = new InputStreamReader(is,"UTF-8");
        //      BufferedReader br = new BufferedReader(ir);
        //      System.out.println(br.readLine());
        //      System.out.println(br.toString().length());
        for (byte b : bytes) {
            System.out.println("该字节是"+b);
            //System.out.println(b==49);
        }
    }
}

in\java.exe" ...

public class Test {
    public static void main(String[] args) {

        Mutiply mutiply = new Mutiply();
        Thread t1 = new Thread(mutiply);
        Thread t2 = new Thread(mutiply);
        t1.start();
        t2.start();
    }
}
2 个用法
class Mutiply implements Runnable{
    2 个用法
    ArrayList<Integer> countList = new ArrayList<Integer>();
    4 个用法
    int count=1000;
    @Override
    public void run() {

        while(count>0){
            System.out.println("第"+count+"次");
            countList.add(count);
            count--;
            System.out.println("循环了"+countList.size()+"次了");
        }
    }
}
```

单元测试通过。

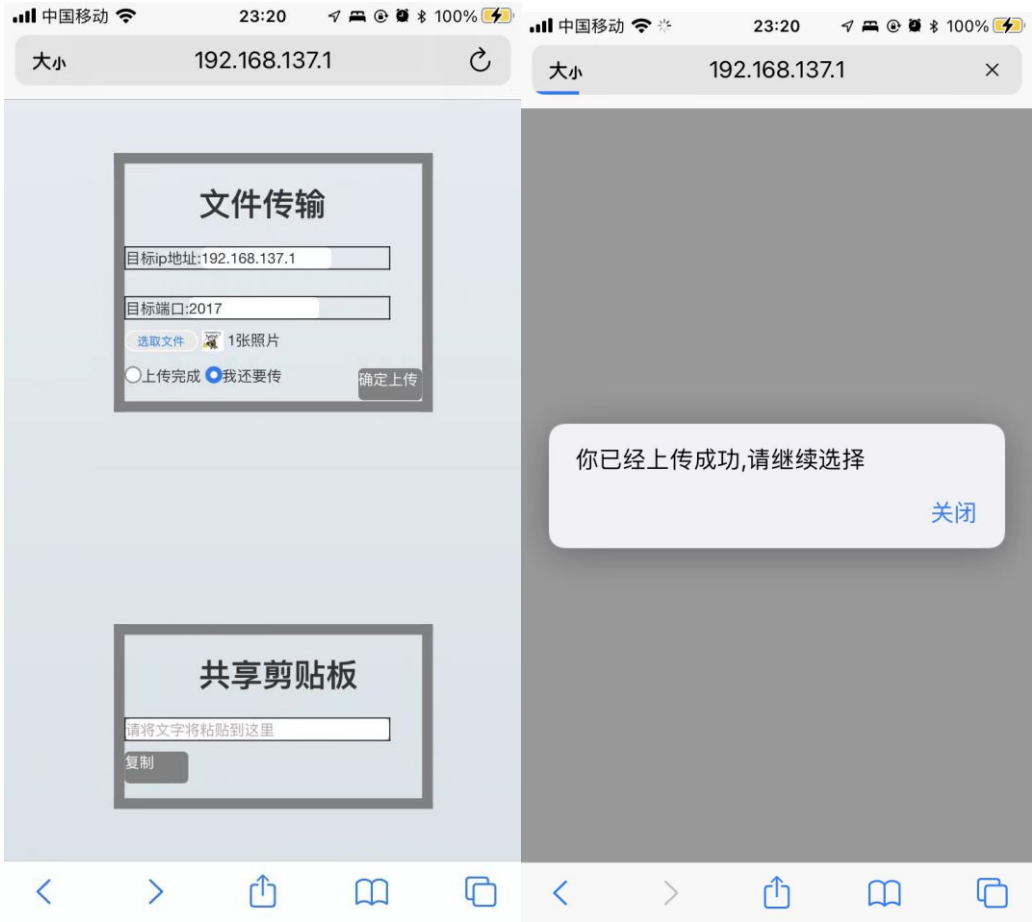
5. 运行效果

运行 Servlet1.java 和 Servlet2.java，完成部署并启动 Tomcat。在手机上，输入服务器网址和端口号，即 192.168.137.1:8080，进入主页面。

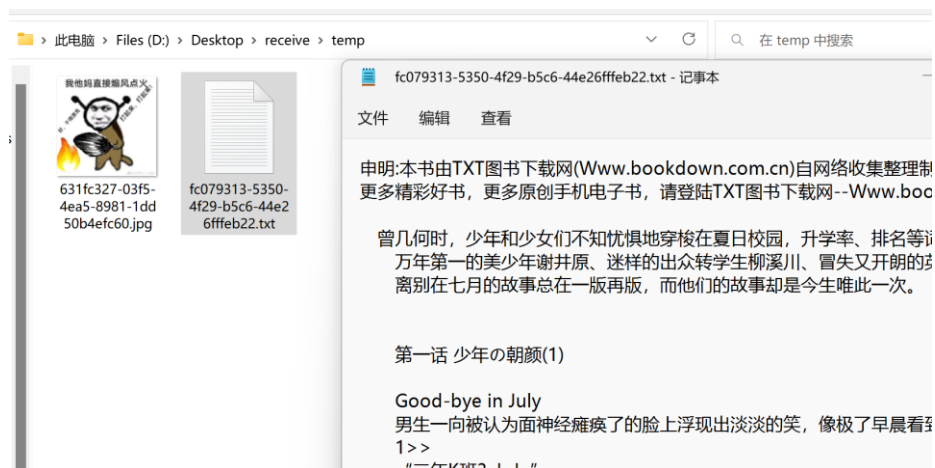


功能 1.1:

输入 IP 地址和端口号，选择文件和是否继续上传，点击确定上传，如下图所示，成功后页面弹出继续上传提示框，即可继续上传。

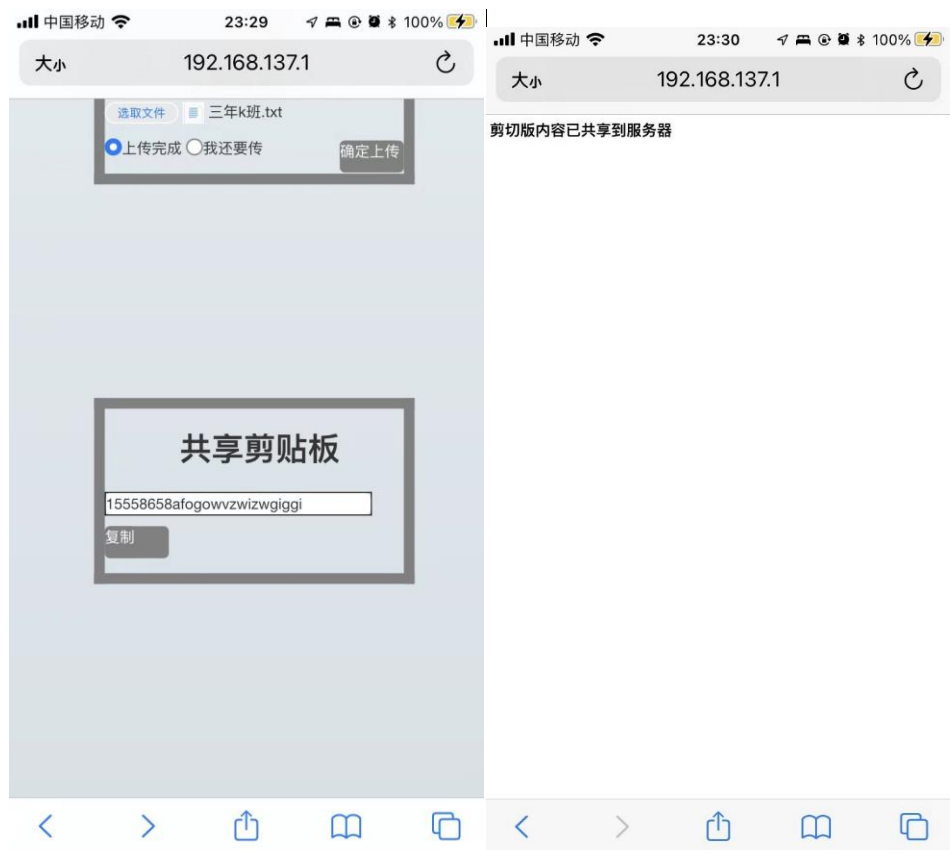


经过两次上传，可从服务器对应位置查看上传的文件。

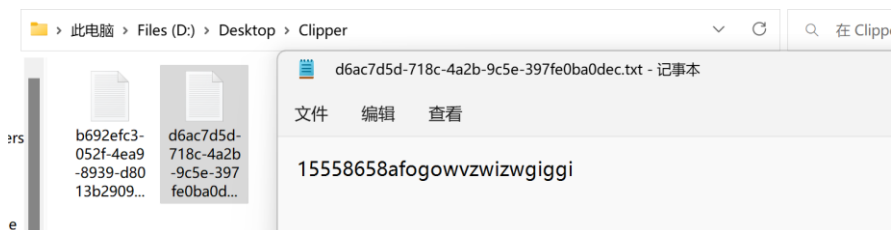


功能 1.2:

输入文字，点击复制，即可实现剪切板功能。



打开文件即可实现查看剪切板内容。



6. 亮点与不足

亮点：

1. 通过对接收的文件的文件名加上一串随机字符，防止上传的文件因重名被覆盖；
2. 为防止大文件上传耗费服务器的资源，用户的文件在上传会进行判断，大文件保存到服务器的 `tmp` 目录下，小文件则保存到 `normal` 目录下；
3. 通过读取 `resource` 目录下的 `config.properties` 获取，服务器管理人员可以根据需要修改，以增加程序的灵活性。

不足：

1. 剪切板传输中文字符串存在乱码问题。
2. 剪切板未实现实时复制粘贴功能，即在手机上复制，在电脑上粘贴。