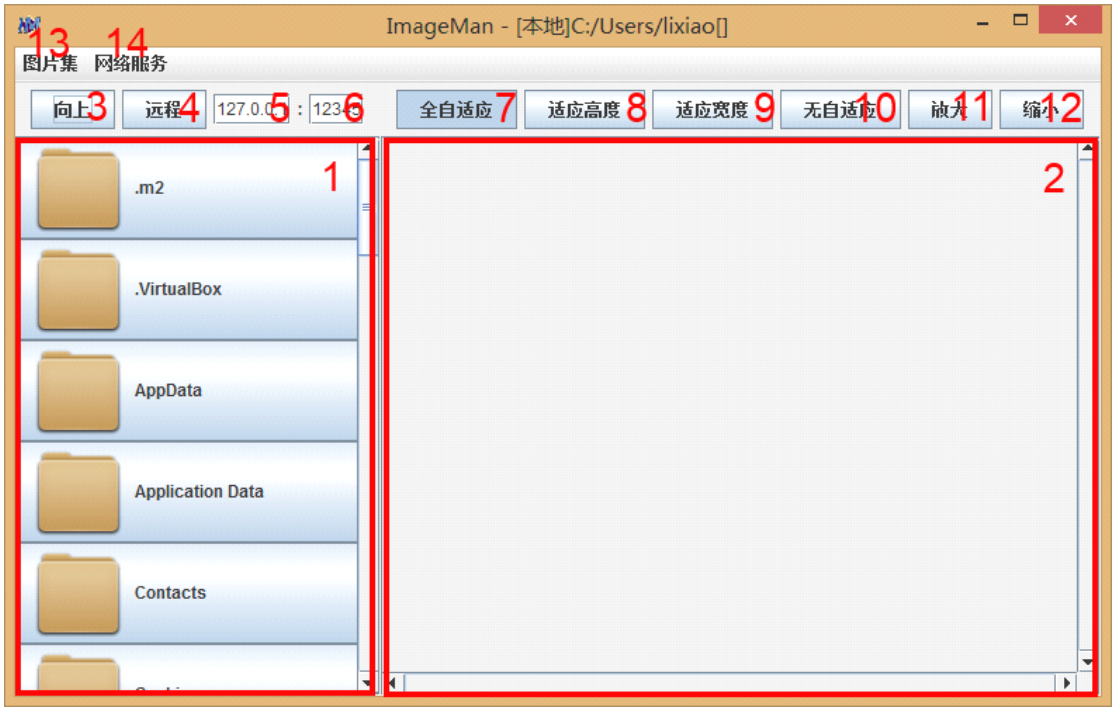
ImageMan 软件文档

10300240007 李霄

2012 年 12 月 31 日

一、操作说明



本程序界面主要分为三部分。左下（1）是文件浏览器，通过鼠标单击可以进入文件夹或打开图片文件，打开的图片会在右下区域（2）的图片查看器中显示。两个区域的上面则是功能按钮区。

（3）为向上按钮，可以在本地文件系统树中向上移动。

（4）为远程连接的开关，打开后可以连接到由地址（5）和端口号（6）指定的服务器，将该服务器共享的目录传输到本地打开。若再将其关闭，则会返回打开之前访问的目录。

（7）、（8）、（9）、（10）为调节图片缩放的互斥开关。（7）打开之后图片会自动缩放，保证显示出全部图像的同时不出现滚动条。

（8）打开之后图片会自动缩放，保证横向滚动条不出现。

（9）打开之后图片也会自动缩放，保证纵向滚动条不出现。

（10）打开之后图片不会自动缩放。

（11）和（12）被点击之后图像自动进入无自适应状态，在无自适应状态下可通过这两个按钮放大或缩小图像。

（13）菜单中有一个选项“进入文件夹”，点击并选择文件夹后会关闭远程连接并进入选择的文件夹。

（14）菜单中有一个选项“启动”，点击后该程序会变为一个服务器，向客户端共享当前的目录，而该选项会变为“停止”，点击后停止服务。

二、功能实现

本程序的类结构及说明如下：

class ManWindow extends JFrame implements ActionListener	主窗体
class ImageThumbView extends JPanel implements ActionListener, Runnable	文件浏览器组件
class ThumbButton extends JButton implements Runnable	文件夹和文件按钮
class ImagePanel extends JPanel implements ActionListener	图片查看器组件
abstract class ImageCollection implements Serializable	文件后端抽象类
class LDCollection extends ImageCollection	本地文件后端
class RemoteCollection extends ImageCollection	远程文件后端
class CollectionServer extends Thread	图片服务器
class CollectionServerThread extends Thread	图片服务器连接线程

1、文件后端的架构

设计这个文件后端的目的是最大限度地扩展文件浏览器的灵活性，让同一个前端可以无缝地浏览本地和远程的文件（本来计划中还能以压缩包作为后端）。这一目的体现在 ImageCollection 的设计中就是用大量的方法实现了对“图片集合”的抽象。

图片集合可以模仿计算机的文件系统被组织成树状结构，每个图片集合拥有返回自己父节点、子节点、图片内容的多种方法。对于本地文件后端，父子节点就是它对应文件夹的父子文件夹，这些方法都通过本地文件操作实现。对于远程文件后端，这些方法则通过与服务器的通信实现。

这样设计的结果就是文件浏览器只需要与“ImageCollection”打交道，任何继承了这个抽象类的类生成的实例都可以成为文件浏览器的后端，不同的后端甚至可以被组织在同一个树中，可以实现丰富的功能。

2、服务器通信的实现

每个程序的实例既可以成为服务器也可以成为客户端。图片服务器在启动之后成为一个独立线程，这个线程每与客户端建立一个连接就会再新建一个连接线程。

通信的协议十分简单，套接字连接建立之后客户端发送字符串请求，请求完毕之后发送一个'\0'，服务器检测到'\0'之后便开始解析请求，然后通过流返回请求的内容。

请求主要分为两类——目录请求和图片请求。二者都是地址字符串，目录请求指向的是目录，服务器返回对应于该目录的 RemoteCollection；图片请求指向的是图片文件，服务器利用 ImageIO 通过流将 bmp 编码的图片内容返回。

三、运行环境

Windows 或 Linux 系统，JRE1.6 以上。

四、小结

本程序一看上去也许亮点并不明显，因为它的创新之处还藏在代码深处，ImageCollection 的抽象是我最为得意的地方。

这次的程序可以说是对我的一个很大的挑战，我给自己定了一个比较高的目标，花了很多时间进行对 ImageCollection 类的设计和修改以及对服务器通信的调试，因此而没有花足够的时间对软件的界面进行设计，但最后的实现依然比较粗糙，不能不说有一些遗憾。如果时间更多一些的话，我应该可以利用现在已经实现的架构为程序添加更多丰富的功能。

然而我在这次试验中也有很多收获，尤其是体会到了真正面向对象设计的优势。活用 Java 语言强大的面向对象功能，可以大大地简化软件开发，让功能的实现更加自由。