GUI Programming with Java

Session 8
Choosers

- We will look at…
  - Using file choosers

File Choosers

- File choosers provide a GUI for navigating the file system.
- They allow a user to choose a file or directory from a list or entering the name of a file or directory.
- To display a file chooser, you usually use the JFileChooser API to show a modal dialog containing the file chooser.
- Another way to present a file chooser is to add an instance of JFileChooser to a container.

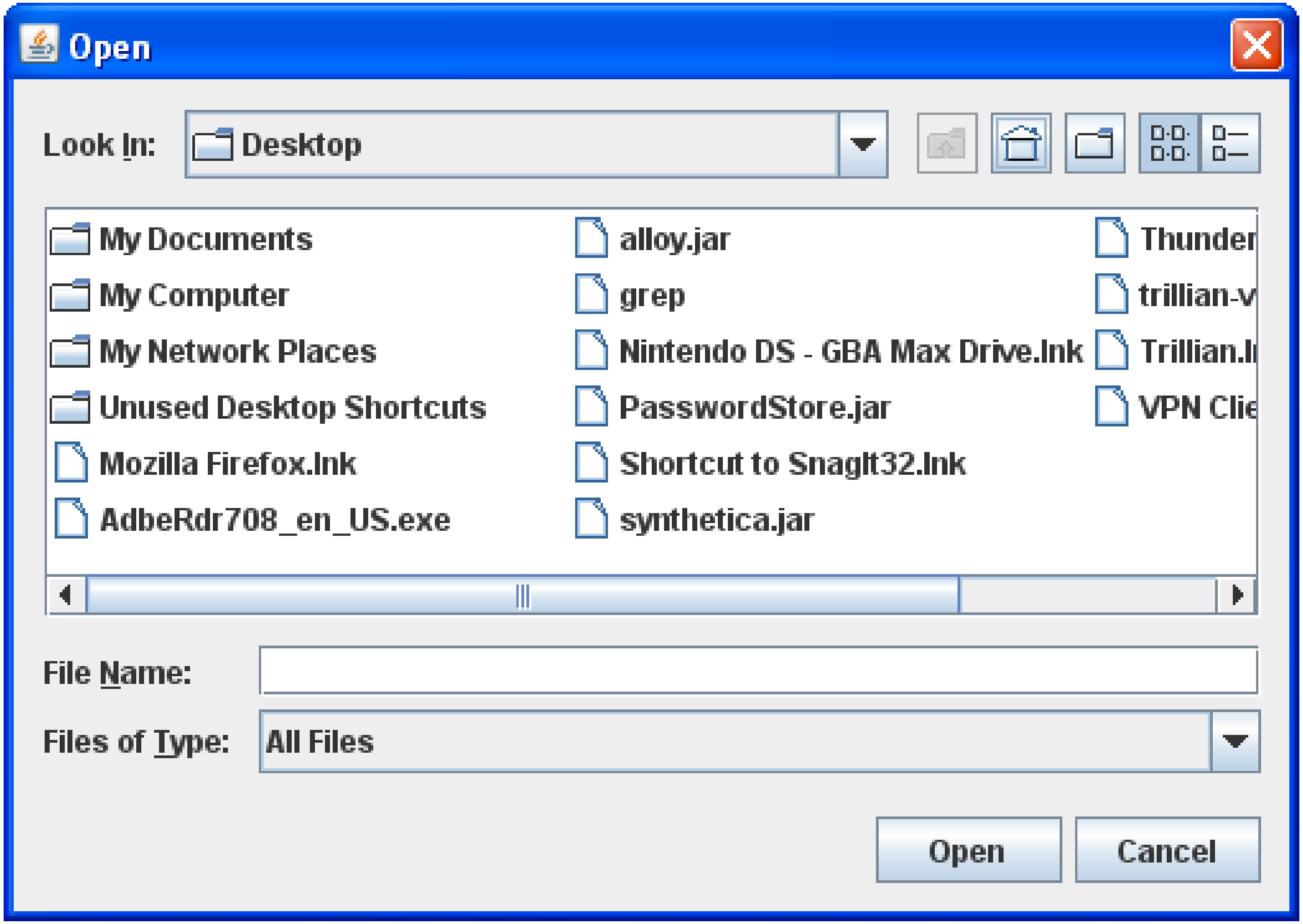
- A JFileChooser object only presents the GUI for choosing files.

- Your program is responsible for doing something with the chosen file, such as opening or saving it.

- The JFileChooser API makes it easy to bring up open and save dialogs.

- The look and feel determines what these standard dialogs look like and how they differ.

- Here is a picture of the Java look and feel's standard open dialog:

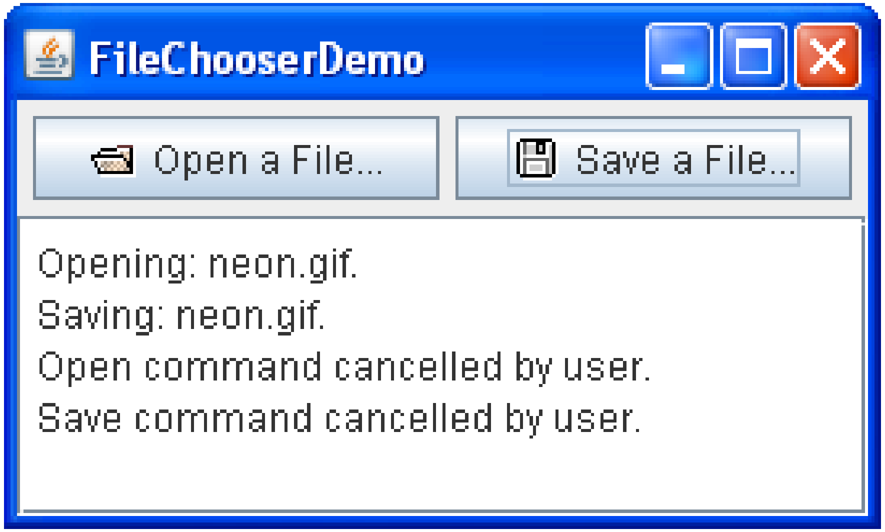


Look In: Desktop

File Name:

Files of Type: All Files

- Lets now take a look at an example program which brings up an open dialog and a save dialog.



```
Opening: neon.gif.
Saving: neon.gif.
Open command cancelled by user.
Save command cancelled by user.
```

- Let’s take a look at FileChooserDemo.java in the sample 1 - file chooser folder.

- Bringing up a standard open dialog requires only two lines of code:

```
//Create a file chooser
final JFileChooser fc = new JFileChooser();
...
//In response to a button click:
int returnVal = fc.showOpenDialog(aComponent);
int returnVal = fc.showOpenDialog(aComponent);
```

- The argument to the showOpenDialog method specifies the parent component for the dialog.
- The parent component affects the position of the dialog and the frame that the dialog depends on.
- For example, the Java look and feel places the dialog directly over the parent component.
- If the parent component is in a frame, then the dialog is dependent on that frame, disappearing when the frame is iconified and reappearing when the frame is deiconified.

Setting the directory

- By default, a file chooser that hasn't been shown before displays all files in the user's home directory. You can specify the file chooser's initial directory using one of JFileChooser's other constructors, or you can set the directory with the setCurrentDirectory method.
- Lets take a look at some of the constructors that may be of use to us.

Constructor Summary

JFileChooser()

Constructs a JFileChooser pointing to the user's default directory.

JFileChooser(File currentDirectory)

Constructs a JFileChooser using the given File as the path.

JFileChooser(String currentDirectoryPath)

Constructs a JFileChooser using the given path.

- The call to showOpenDialog appears in the actionPerformed method of the **Open a File...** button's action listener:

```
    public void actionPerformed(ActionEvent e) {
    //Handle open button action.
    if (e.getSource() == openButton) {
    int returnVal = fc.showOpenDialog(FileChooserDemo.this);
        if (returnVal == JFileChooser.APPROVE_OPTION) {
        File file = fc.getSelectedFile();
        //This is where a real application would open the file.
    log.append("Opening: " + file.getName() + "." + newline);
        }
        else {
              log.append("Open command cancelled by user." + newline); }
        }
... }
```

Using the file object

- Remember that last semester you studied File IO which would have included a look at the File class.

- You can call other methods on the File object, such as getPath, isDirectory, or exists to get information about the file.

- You can also call other methods such as delete and rename to change the file in some way.

- Of course, you might also want to open or save the file using one of the reader or writer classes provided by the Java platform.

Forgotten IO?

I suggest checking your notes or visiting

http://java.sun.com/docs/books/tutorial/essential/io/index.html

- The example program uses the same instance of JFileChooser to display a standard save dialog. This time the program calls showSaveDialog:

```
int returnVal = fc.showSaveDialog(FileChooserDemo.this);
```

By using the same file chooser instance to display its open and save dialogs, the program reaps these benefits:

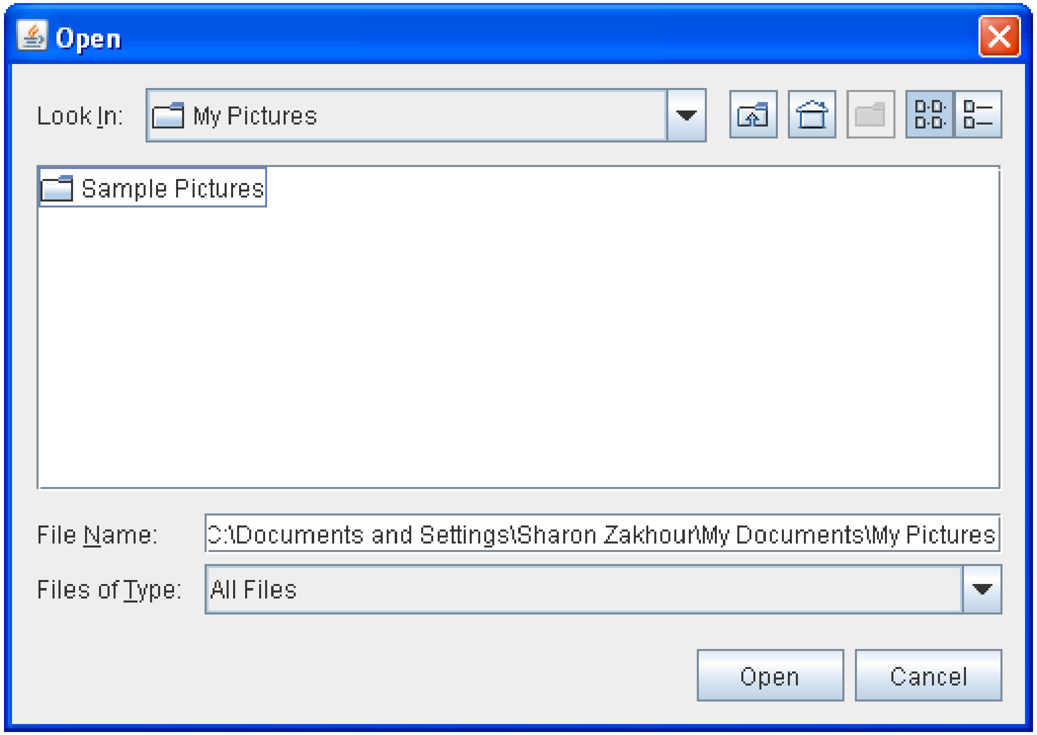
  - The chooser remembers the current directory between uses so the open and save versions automatically share the same current directory.
  - You have to customize only one file chooser, and the customizations apply to both the open and save versions of it.

- Finally, the example program has commented-out lines of code that let you change the file selection mode.

- For example, the following line of code makes the file chooser able to select only directories, and not files:

```
fc.setFileSelectionMode(JFileChooser.DIRECTORIES_ONLY);
```

- Another possible selection mode is FILES_AND_DIRECTORIES. The default is FILES_ONLY.
- The following picture shows an open dialog with the file selection mode set to DIRECTORIES_ONLY.
- Note that, in the Java look and feel at least, only directories are visible — not files.



File Name: C:\Documents and Settings\Sharon Zakhour\My Documents\My Pictures

Files of Type: All Files

- Exercise for this week

Create a program that a Java Jframe program that launches a JfileChooser. You need not worry about implementing the code to modify files, just instantiate the chooser and explore with it.

Get started on assignment 2

END OF SESSION