

# Deployment: JARs, Applets, and Web Start

## Lecture 30

## The Problem

- A Java application consists of
  - Classes, interfaces, exceptions (ie files)
    - One of these is special, contains main
  - Packages (ie directories)
  - Other files (text, images for icons, etc)
- How do you submit your lab in Carmen?
  - Zip/tar everything into one file
- How do we run your lab?
  - Unzip/untar the submission
  - Compile everything
  - Find the class that contains main
  - Run the application

```
$ java JottoGame
```
- Is there an easy way to deploy an app?
  - No unzipping, compiling, looking for main

## Solution #1: JAR files

- A Java Archive (JAR) file is the tar+zip of class files, directories, and other files
  - A single file
    - Easy for client to manage
  - JVM can access contents of JAR directly
    - No need for client to explicitly untar+unzip
    - No need for client to compile
  - Includes name of main class (optional)
    - No need for client to know name of class
- To create from command line

```
$ jar cvf Calc.jar *.class icon.gif
```

## Eclipse Demo: CalcMVC

- Create JAR
  - Basic: Export > Java > JAR file
    - Select packages and files to include
    - Identify main class (optional)
  - Alt: Export > Java > Runnable JAR file
    - Select a launch configuration
- Run application
  - Basic: From the command line

```
$ java -jar Calc.jar
```
  - On Solaris, run Calculator.jar directly

```
$ Calc.jar
```
  - On Windows/Mac, double-click on jar file

## Locating Resources

- "Resource": A file used by a Java app
  - Eg Word dictionary for Jotto game
- Goal: A robust way to find the file
- Solution: use java.lang.Class

```
URL url = JottoModel.class.getResource("icon.gif");
InputStream s =
    JottoModel.class.getResourceAsStream("w.txt");
```
- Location is relative to location of JottoModel.class file
  - "words.txt": in same directory
  - "data/words.txt": in subdirectory data/
  - "/data/words.txt": independent of location of JottoModel.class (ie is relative to classpath)

## Solution #2: Applets

- Another way to deliver applications: over the web!
  - Client points a browser at a site containing an applet
  - Bytecode is shipped over the network
  - Applet runs in client's browser
    - Browser has a JVM
- Two parts:
  - Compiled Java code (including a JApplet)
  - An html file directing the browser to create and run the JApplet

## The Life Cycle of a JApplet

- JApplet is a top-level container (like JFrame)
  - Plus: init, start, stop, destroy
  - Minus: setSize, setDefaultCloseOperation, setTitle, setVisible
- When page is first loaded:
  - The JApplet is instantiated
  - First init() called, then start()
- When browser leaves/returns to page:
  - stop() called when leaving
  - start() called when returning
- When browser exits
  - stop() called, then destroy()

## Trivial Example

- HTML file

```
<applet code="Trivial.class"
width="400" height="50">
</applet>
```
- Java code

```
public class Trivial extends JApplet {
    private String msg = "Ciao";
    public void init() {
        add(new JLabel(msg));
    }
}
```

## Applet Features

- Obtaining parameters from web page

```
<applet code="Trivial.class"
width="400" height="50">
    <param name="quote" value="Hello" />
</applet>
```

  - In JApplet, can read name/value

```
String msg = getParameter("quote");
```
- Combining with JARs

```
<applet code="CalcView.class"
codebase="myApplets"
archive="Calc.jar"
width="300" height="100">
</applet>
```

## Translating Swing App to Applet

- Make HTML page with <applet> tag
- Eliminate main
- Subclass JApplet (must be public) instead of JFrame
  - Move code from constructor to init()
- Remove calls to
  - setSize (JApplets are sized by applet tag)
  - setDefaultCloseOperation
  - setTitle (JApplets do not have title bars)
  - setVisible
- See CalcApplication vs CalcApplet
  - Demo: add applet tag to web page
  - Demo: create JAR and make it web-available
- Advice: write GUI view as a JPanel, which you can then put in either a JFrame or a JApplet

## Debugging Applets

- Java Console is helpful for tracing
  - Should appear whenever an applet loads
  - Windows: Control Panel > Java Plug-in
  - Linux: \$ jcontrol
- Beware: Applet code is cached!
  - Separate from browser cache, history, etc
  - Clear applet cache from Java console
    - Command: x

## Solution #3: Web Start

- Alternative to applets
  - Applets are old-school
- Still delivered via a browser
- But it does not execute inside the browser
  - Displayed in its own window, stand-alone
- Does not use browser's JVM
- Supports digital signatures
  - Unsigned apps execute in a sandbox

## Web Start: Steps

Computer Science and Engineering @ The Ohio State University

- Produce a JAR of the application, Calc.jar
- Configure web server to deliver this particular application
  - Create a deployment directory
    - eg tomcat/webapps/calculator
  - Create subdirectory WEB-INF, containing a basic web.xml file
    - eg tomcat/webapps/calculator/WEB-INF/web.xml
- Prepare a "launch file", Calculator.jnlp
  - An XML file that defines (1) the codebase (2) the name of the JAR
- Put JNLP in the deployment directory
  - Calculator.jnlp is web-visible
- Put JAR in the codebase (see launch file)
- Point browser to JNLP file

## Summary

Computer Science and Engineering @ The Ohio State University

- JAR files
  - Single file archive of entire application
  - Locating resources relative to .class files
- Applets
  - Application delivered by a web server
  - Executes inside a web page
  - Use JApplet instead of JFrame
  - Lifecycle: init, start, stop, destroy
- Web Start
  - Application delivered by web server
  - Executes outside of browser