Lecture 25

Java Beyond DrJava

Announcements for This Lecture

This Week

- Reading: Chapter 16
- Assignment A6 graded
  - Mean: 86.8, Median: 90
  - Mean: 10.5h, Median: 10h
- No new lab this week
  - Turn in lab from last week
  - Work on assignment A7
- Assignment A7 due Saturday

Next Week

- **Submit a course evaluation**
  - Will get an e-mail for this
  - Part of the "participation grade" (e.g. clicker grade)
- **Final, May 10th 9:00-11:30**
  - Review posted later this week
- **Conflict with Final Exam?**
  - e.g. > 2 finals in 24 hours
  - Submit conflicts on CMS

Announcements for This Lecture

This Week

- Reading: Chapter 16
- Assignment A6 graded
  - Mean: 86.8, Median: 90
  - Mean: 10.5h, Median: 10h
- No new lab this week
  - Turn in lab from last week
  - Work on assignment A7
- Assignment A7 due Saturday

Next Week

- **Review sessions next week**
  - Still lining up times
  - 3 sessions/day in 1 hour slots
  - Monday, Tuesday 1-4
  - Either Sunday or Wednesday
- **Topics posted Thursday**
- **Conflict with Final Exam?**
  - e.g. > 2 finals in 24 hours
  - Submit conflicts on CMS

Steganography Observation

- Most you preferred end markers to using length
- But few cons to length
  - Only requires two pixels (e.g. <= 999,999)
  - Hard part: conversion
- Markers okay if not *printable*
  - Non-printable chars: <= 32
  - Or 3-digits numbers > 255
- Bad if marker is in message
  - `reveal` will terminate early

message terminates

```
/**
 ...
* Format: ## message ##
*/
public ImageProcessor {
  ...
}
```

Tried to "hide" your source code

Java Outside the Interactions Pane

- Every Java program is either an application or an applet.

```
public class C {
    ...
    public static void main(String[] args) {
        // top method to invoke
        ...
    }
    ...
}
```

The parameter, an array of Strings, is used to pass information to the program

- **Application**: class with a special static method (main)
- Run the application by invoking this method
  - Interactions pane
  - OS command line
  - Double-clicking on it?

Executing Java from Command Line

Java Code

```
public class C {
    ...
    public static void main(String[] args) {
        // top method to invoke
        ...
    }
    ...
}
```

Command Line

> cd <folder>

(moves to that folder)

> dir (Windows) or ls (OS X

(list of files)

> java C

(executes C.main(null))

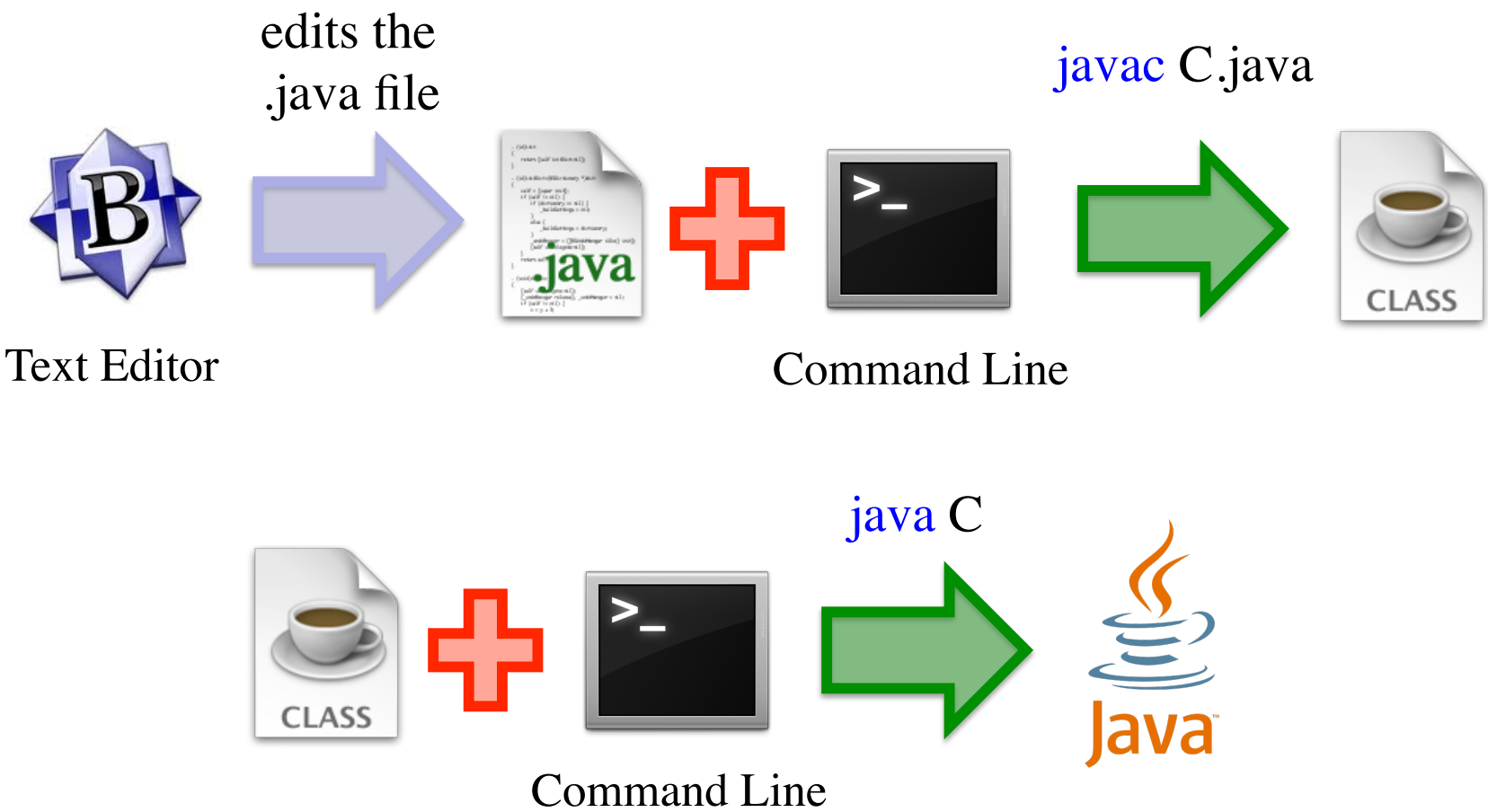
Can type in Interactions page

“Simplest” Java Application

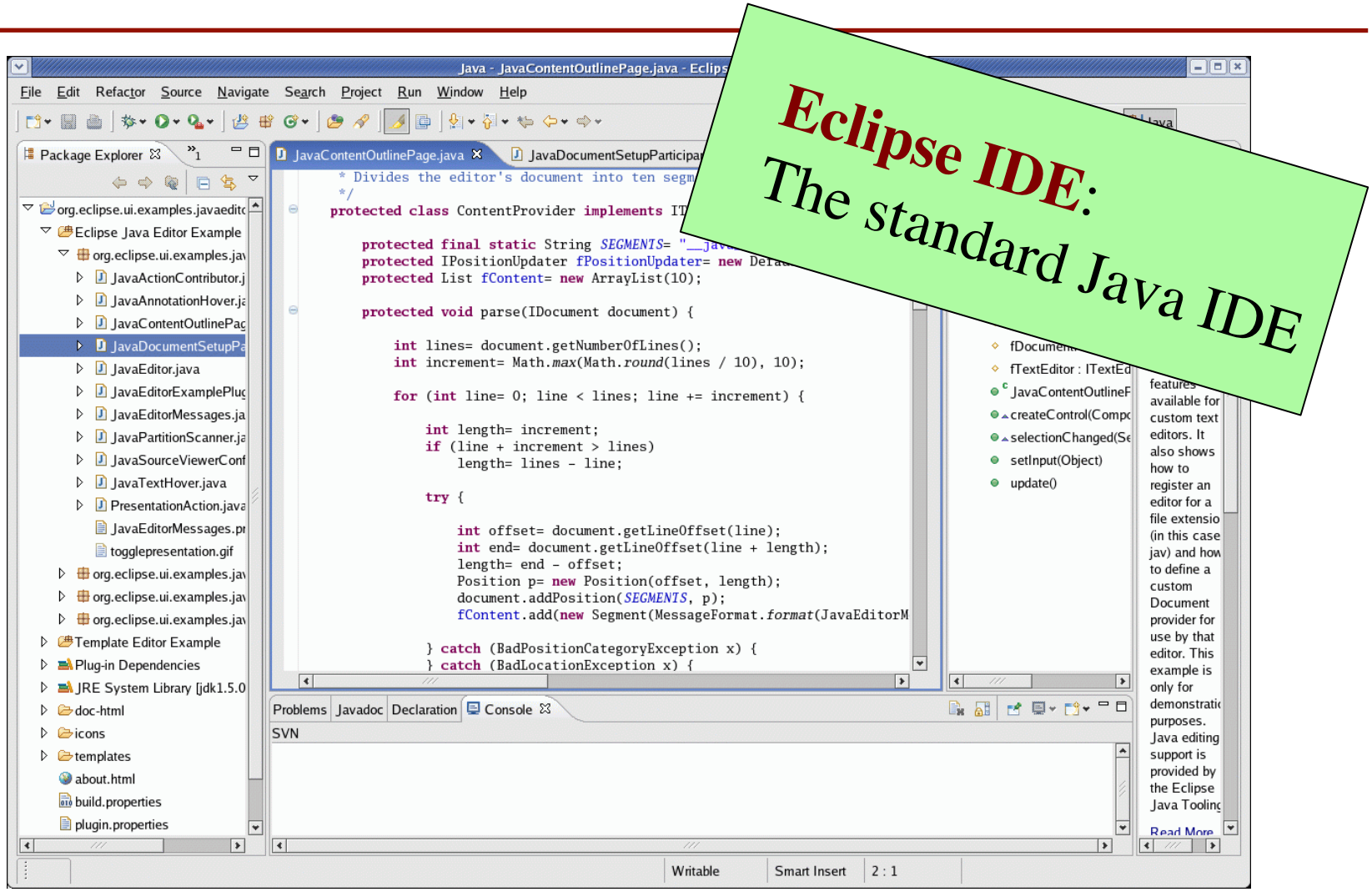
```
public class Simple {
   public static void main(String[] args) {
        System.out.println(“Hello World”)
   }
}
```

Execute with “java Simple”

Writing a Java Application: Classic Way



Applications: A Slightly Harder Way



To Use an IDE or Not?

Demo Time!

Advantages

- Organize all your classes
  - MVC needs multiple classes
  - Organize them as a "Project"
- Auto-generated code
  - GUI design
  - API auto-completion
- Interactive debugging
  - Breakpoints
  - Variable watches

Disadvantages

- Overwhelming!

- Sometimes you just want a single, simple class
  - No Projects
  - No "workspaces"

Java JAR Files

- Goal: “double-clickable” app
- JAR: Java Archive File
  - Compressed file collection
  - Similar to a ZIP file
  - Except it can be executed
- Jar files contain
  - All the necessary class files
  - Any image or sound files
  - Any other necessary files
  - A **manifest** file

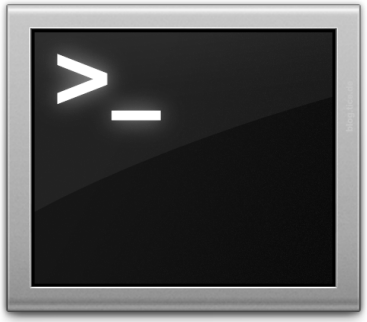
- manifest: *noun*
  - list of passengers
  - invoice of cargo
- Identifies the class with `main`
  - Might have more than one

Executing a Java File

- Double-click it!

- Supported in most OSs
  - But error if no manifest

- Command line



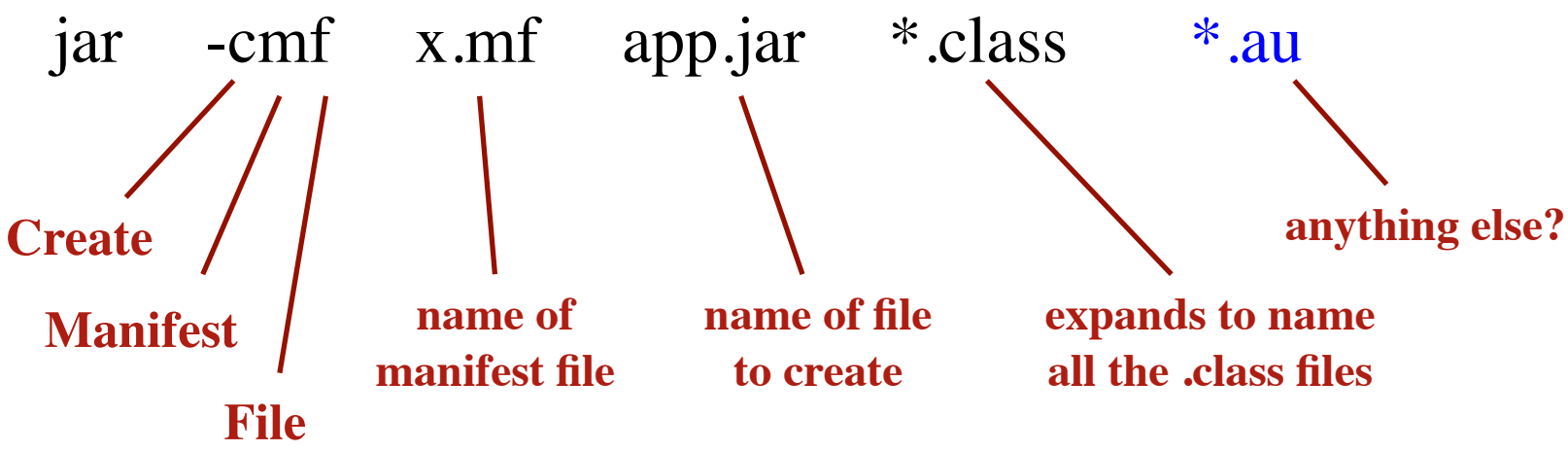
- Type:
  java -jar <jar-file>

Creating a JAR File

1. Navigate to the directory that contains the .class files.
2. Create a text file x.mf with one line (ending in a line-feed):

   Main-class: <name of class>

3. In the directory, type:



Inspecting JAR File Contents

- List files in a Jar file:

jar -tf images.jar

type (list)

File

name of the jar file

```
> jar tf acm.jar
acm/graphics/
acm/graphics/G3DRect.class
acm/graphics/ArcRenderer.class
acm/graphics/GArc.class
acm/graphics/GMouseEvent.class
acm/graphics/GCanvasListener.class
acm/graphics/GCanvas.class
acm/graphics/GCompound.class
acm/graphics/GIterator.class
acm/graphics/GContainer.class
acm/graphics/GDimension.class
acm/graphics/GFillable.class
…
```

Applets vs. Applications

```
public class C {
    public static void main(String[] args) {
    ...}
}
```

application

```
import javax.swing.*;
public class A extends JApplet {
   public void init() { ... }
   public void start() { ...}
   public void stop() { ... }
   public void destroy() { ... }
}
```

applet

- **Applet**: Java program run in a web browser
  - Needs an html page

Four inherited procedures:

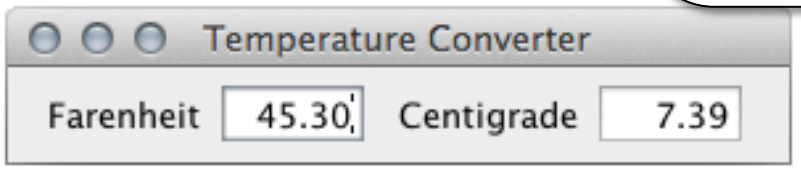
- called to initialize
- called to start processing
- called to stop processing
- called to destroy resources (just before killing the applet)

TemperatureConverter Example

Application

Latest Version can be both!

Applet




```
private void initAsApplication() {
  JFrame frame =
      new JFrame("Temperature Converter");
  frame.setDefaultCloseOperation(
              JFrame.EXIT_ON_CLOSE );
  frame.getContentPane().add(view);
  frame.pack();
  frame.setVisible(true);
}
```

```
public void init() {
    getContentPane().add(view);
}

public void start() { /* Do nothing */ }

public void stop()  { /* Do nothing */ }

public void destroy() { /* Do nothing */ }
```

An Applet HTML Page

```
<html>
    <head>
        <title>FacultyApplet</title>
    </head>
    <body>
    <h2>This is an <i>Applet!</i></h2>
    <p>
    <applet archive="temperature.jar"
        code="converter.TemperatureConverter"
        width="600"    height="100">
    </applet>
    </p>
  </body>
</html>
```

tags

<html>	start an html page
<head>	start the "heading"
<title>	the title for the page
<body>	start the body, content, of the page
<h*x*>	begin heading level x
<p>	begin a paragraph
<b>	begin boldface
<i>	begin italics
<applet>	start a Java applet

What Happened to Applets?

The Browser Wars

- Java supported as "plug-in"
  - Java controlled by Sun (now Oracle)
  - Browsers made by 3rd party
- Could not ensure up to date
  - Install is harder than Flash
  - Requires OS-level access
  - Think about your install!
- People no longer bothered
  - Applets almost non-existent

Modern Day Web

- Browsers support Javascript
  - Very different language!
  - But is what Java "promised"
  - Name for marketing reasons
- Java is used on the back-end
  - e.g. code on the servers
- GWT: Google Web Toolkit
  - Java for browser & server
  - Browser side code compiles to JavaScript (can do that!)