# Applets

# Outline

- Basics of applets
- First applet – HelloWorldApplet.java
- More on applets
- Animation applet – digital clock

# What Are Applets?

- An *applet* is a special Java program that can be embedded in HTML documents.
- It is automatically executed by (applet-enabled) web browsers.
- In Java, non-applet programs are called *applications*.

# Application vs. Applet

- Application
  - Trusted (i.e., has full access to system resources)
  - Invoked by Java Virtual Machine (JVM, `java`), e.g., `java HelloWorld`
  - Should contain a main method, i.e., `public static void main(String[])`
- Applet
  - Not trusted (i.e., has limited access to system resource to prevent security breaches)
  - Invoked automatically by the web browser
  - Should be a subclass of class `java.applet.Applet`

# Examples

- HelloWorld.java

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

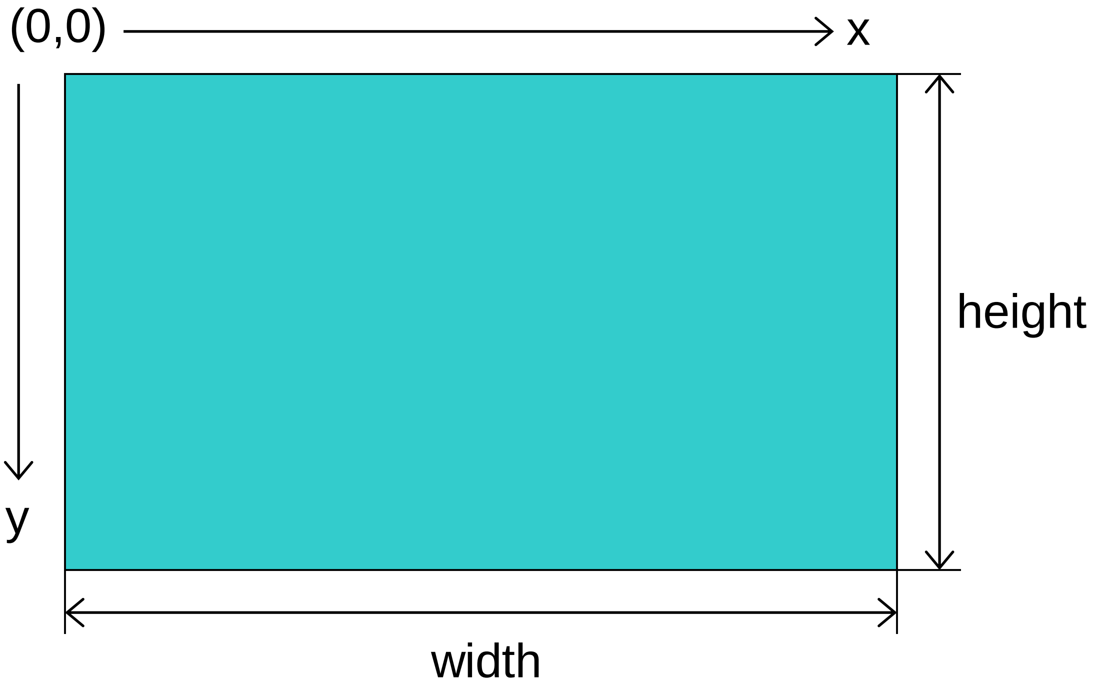
- HelloWorldApplet.java

# First Java Applet

- Java source in `HelloWorldApplet.java`

```
import java.awt.*;
import java.applet.Applet;
public class HelloWorldApplet extends Applet {
    public void paint(Graphics g) {
        Dimension d = getSize();
        g.setColor(Color.BLACK);
        g.fillRect(0, 0, d.width, d.height); // paint background
        g.setFont(new Font("San-serif", Font.BOLD, 24));
        g.setColor(new Color(255, 215,0));
        g.drawString("Hello, world!", 60, 40);
        g.drawImage(getImage(getCodeBase(), “Rabbit.jpg"),
           20, 60, this);
    }
}
```

# Graphics Coordinate



# Embedding Applet into HTML

- HTML source in `HelloWorld.html`

```
<!--HelloWorld.html-->
<html>
  <head>
    <title>HelloWord</title>
  </head>
<body>
 <center>
 <applet code="HelloWorldApplet.class"
     width=300 height=350></applet>
</center>
<hr/>
<a href="HelloWorldApplet.java">The source.</a>
</body>
</html>
```

# Compiling and Running

- To compile

  ```
  javac HelloWorldApplet.java
  ```

  Produces `HelloWorldApplet.class`

- To run
  - Open page `HelloWorld.html`from web browser or
  - Use appletviewer of JDK

  ```
  appletviewer HelloWorld.html
  ```

# Elements of Applets

- Superclass: `java.applet.Applet`
- No `main` method
- `paint` method to paint the picture
- Applet tag: `<applet> </applet>`
  - `code`
  - `width` and `height`

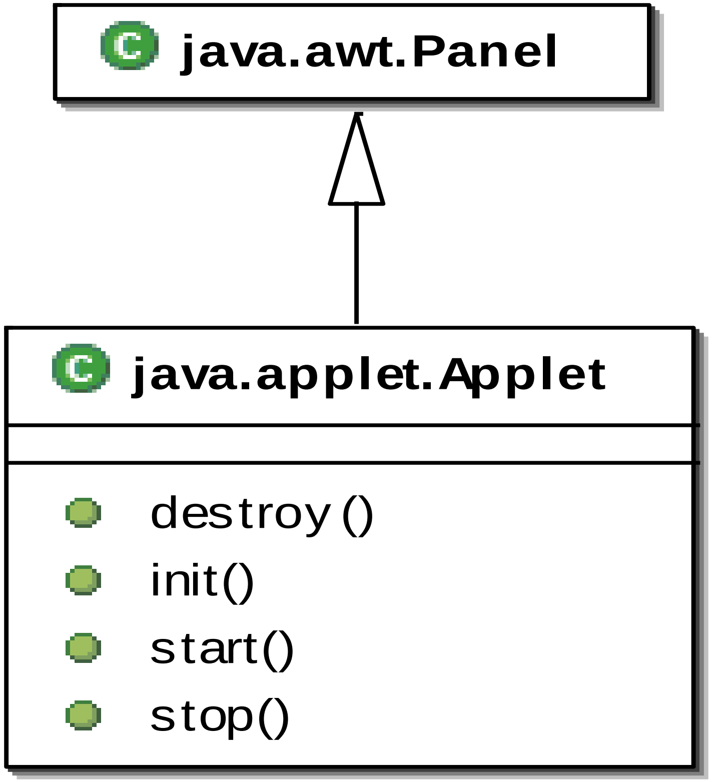
# Outline

- ✓ Basics of applets
- ✓ First applet – HelloWorldApplet.java
- ○ More on applets
- ○ Animation applet – digital clock

# Framework-Based Programming

- (OO) Frameworks
  - Semi-complete applications.
  - Provide the structure (backbone) and utilities for applications.
  - Often domain specific.
  - *Inversion of control.*
  - Examples: applets, GUI frameworks, etc.

# The Class Applet



# The Life-Cycle of Applet

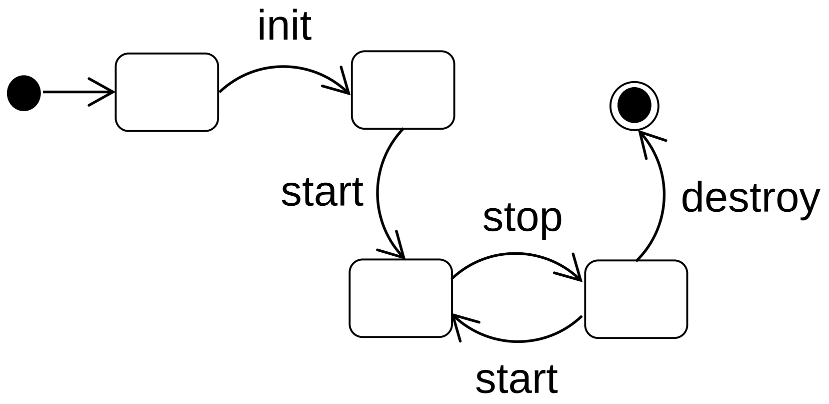
- init()
  - Called exactly once in an applet’s life.
  - Called when applet is first loaded, which is after object creation, e.g., when the browser visits the web page for the first time.
  - Used to read applet parameters, start downloading any other images or media files, etc.

# Applet Life-Cycle (Cont.)

- start()
  - Called at least once.
  - Called when an applet is started or restarted, i.e., whenever the browser visits the web page.
- stop()
  - Called at least once.
  - Called when the browser leaves the web page.

# Applet Life-Cycle (Cont.)

- destroy()
  - Called exactly once.
  - Called when the browser unloads the applet.
  - Used to perform any final clean-up.



# Outline

- ✓ Basics of applets
- ✓ First applet – HelloWorldApplet.java
- ✓ More on applets
- ○ Animation applet – digital clock

# Animation Applet --- Digital Clock

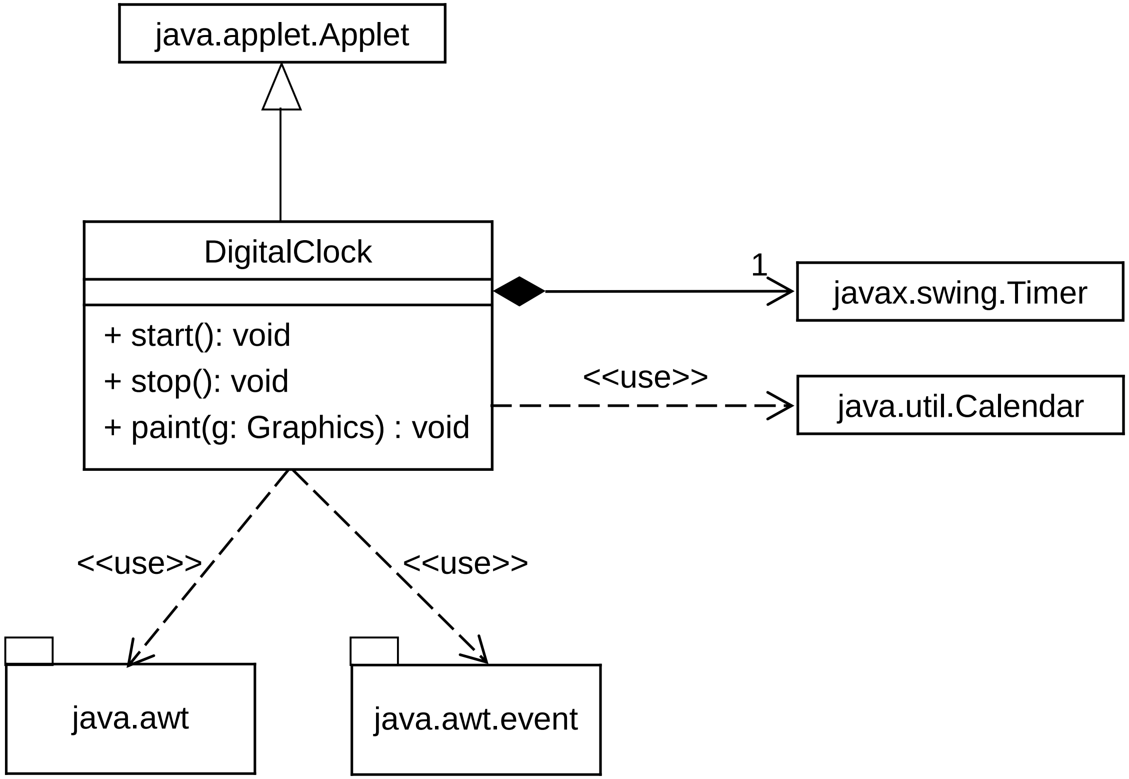
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.Calendar;

public class DigitalClock
    extends java.applet.Applet {

    <Fields>
    <Methods>
}
```

An applet must be a subclass of `java.applet.Applet`.

# Program Structure



# Methods for Digital Clock

- `public void start(){...}`
  invoked when entering the web page that contains the applet
- `public void stop(){...}`
  invoked when leaving the web page that contains the applet
- `public void paint(Graphics g){...}`
  paint the picture

# Field Declarations

```
protected Timer timer;
protected Font font =
    new Font("Monospaced", Font.BOLD, 48);
protected Color color = Color.GREEN;
```

# Object Initialization

```
public DigitalClock() {
        timer = new Timer(1000, createTimerTickHandler());
 }

protected ActionListener createTimerTickHandler() {
        return new ActionListener() {
                public void actionPerformed(ActionEvent event) {
                    repaint();
                }
            };
}
```

# The `start()` and `stop()` Methods

```
public void start() {
  timer.start();
}

public void stop() {
  timer.stop();
}
```

- Start and stop the timer
- Stopped timer will not consume CPU time.

# The `paint()` Method

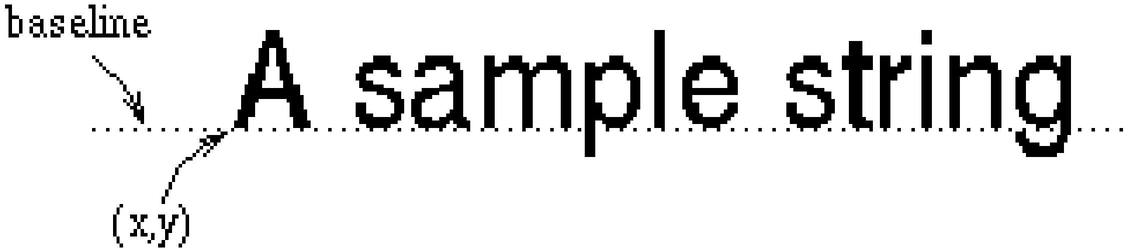
```
public void paint(Graphics g) {
  Calendar calendar = Calendar.getInstance();
  int hour = calendar.get(Calendar.HOUR_OF_DAY);
  int minute = calendar.get(Calendar.MINUTE);
  int second = calendar.get(Calendar.SECOND);
  g.setFont(font);
  g.setColor(color);
  g.drawString(hour / 10 + hour % 10 +
        ":" + minute / 10 + minute % 10 +
        ":" + second / 10 + second % 10,
        10, 60);
}
```

# Who Calls the `paint()` method?

- `Timer` ticks and calls `ActionListener.actionPerformed()`
- `ActionListener.actionPerformed()` calls `DigitalClock.repaint()`
- `DigitalClock.repaint()` calls `DigitalClock.paint()`
- The `paint()` method is usually not called directly.

# Drawing Strings

```
g.drawString("A sample string", x, y)
```



# HTML Source

```
<!-- DigitalClock.html -->
<html>
  <head>
    <title>Digital Clock Applet</title>
  </head>
<body bgcolor=black>
  <h1>The Digital Clock Applet</h1><p>
  <applet code=DigitalClock.class width=250 height=80>
  </applet>
  <p><hr>
  <a href=DigitalClock.java>The source</a>
</body>
</html>
```

# The `java.awt.Color` Class

- Instances of the `Color` class represent colors.

```
new Color(r, g, b)
```

where *r*, *g*, *b* are the values of the red, green, and blue components, respectively. They are in the in the range of 0 to 255.

- Predefined constants

```
BLACK ORANGE YELLOW BLUE GREEN PINK
CYAN LIGHTGRAY RED ARKGRAY MAGENTA
WHITE
```

# The `java.awt.Font` Class

- Fonts are specified with three attributes:
  - *font name*:
    `Serif Sans-serif Monospaced Dialog`
    `DialogInput TimesRoman Helvetica Courier`
  - *font style*:
    `PLAIN BOLD ITALIC`

    Styles can be combined: `Font.BOLD|Font.ITALIC`
  - *font size*: a positive integer
- A font can be created as follows:

`new Font(`*name*`,` *style*`,` *size*`)`

# Exercise

- Write an applet class named CuckooDigitalClock that behaves like DigitalClock except that it cuckoos at every hour by playing an audio file “cuckoo.au”.

  Hints

  - To play an audio file, use play(getCodeBase(), “cuckoo.au”).
  - To get the current time, use the class java.util.Calendar, e.g.,

    ```
    Calendar calendar = Calendar.getInstance();
    int hour = calendar.get(Calendar.HOUR_OF_DAY);
    int minute = calendar.get(Calendar.MINUTE);
    int second = calendar.get(Calendar.SECOND);
    ```