Overview

- AWT Basic
- Use AWT
- AWT Event
- Specific things
What is the AWT?

- Classes which have a collection of common graphic functions supporting several GUI platforms
- A Classical method of windows programming
  - Based on system executing
  - heavyweight component
  - java.awt.* You have to import this
- Function
  - Control and manage graphic contexts
  - Draw shapes and text
  - Control and manage images

- So use Swing
- Swing is light component
Why we have to know AWT?

- Two kinds of user interface
  - CUI (Character User Interface)
  - GUI (Graphical User Interface)

- Many programs are based on GUI
  - Windows, Media Player, Editplus, games etc.

- In Java, you can programming GUI using AWT and Swing
AWT Components

- Button
- Canvas
- Checkbox
- Choice
- Container
- Label
- List
- Scrollbar
- TextComponent
  - TextArea
  - TextField
- Panel
- ScrollPane
- Dialog
- Window
- Frame
- OpenFileDialog
How to use Components

1. Generate a component
   Button myButton = new Button("my button");

2. Add component on Container
   add(myButton);

3. Connect event listener and event control routine
   MyActionListener mal = new MyActionListener();
   myButton.addActionListener(mal);
Overview

- AWT Basic
- Use AWT
- AWT Event
- Specific things
Event

- Event
  - Response about user’s behavior about UI components

- Programming based on event
  - Using busy waiting, watch user’s behavior and response about that.
Event Handling

- `java.awt.event`
  - Provide interfaces and class for handle several events from AWT component
  - `xxxListener`, `xxxEvent`, `xxxAdapter` Interface Class

- Delegated event handling model
  - Register object(event listener) at Component(event source) for event handling
## Kinds of event

**Low-Level event and Semantic event**

<table>
<thead>
<tr>
<th>Low-Level event</th>
<th>Semantic event</th>
</tr>
</thead>
<tbody>
<tr>
<td>System level events resulted from User’s input or component’s function</td>
<td>Using Low-Level event, make a second event (e.g) mouse click -&gt; pressed, released 2 kinds of Low-Level event occur =&gt; Handled by ActionEvent</td>
</tr>
</tbody>
</table>

**Kinds of Low-Level event**
- java.awt.event.ComponentEvent
- java.awt.event.FocusEvent
- java.awt.event.KeyEvent
- java.awt.event.MouseEvent
- java.awt.event.WindowEvent

**Kinds of Semantic event**
- java.awt.event.ActionEvent
- java.awt.event.AdjustmentEvent
- java.awt.event.ItemEvent
- java.awt.event.TextEvent

In practical event handling, these are not distinguished Handling Semantic event is more efficient
Step for calling event listener

- **processEvent**
  - When event occurs at component, call event listener

```
processEvent => processFocusEvent => FocusListener, and process it based on ID
```
Event class and Event listener

- Classify kinds of event that can be occur at component
  - Event classes have information and method about event
- Fixed event at each components
  - Button event
    - ActionEvent, ComponentEvent, FocusEvent, KeyEvent, MouseEvent
  - Components that ActionEvent can be occur
    - Button, List, MenuItem, TextField
  - Etc, referencing API
Hierarchy of event classes
Event class and Event listener

- **Listener** is interface handling events
  - xxxEvent class maps onto xxxListener
- xxxListener have method handling low-level event
  - E.g – KeyListener handling KeyEvent
    - void keyPressed(KeyEvent e)
    - void keyReleased(KeyEvent e)
    - void keyTyped(KeyEvent e)
Event handling process

- Decide the event to handle
  - E.g. ActionEvent

- Define a class which include event listener
  - E.g. MyListener is a class implementing ActionListener
    ```
    class MyListener implements ActionListener{
    ...
    }
    ```
    - Implement void actionPerformed(ActionEvent e)
  - Each listener has different implement (referencing API)

- Generate an object and register it as event listener
  - this.addActionListener(new MyListener());
Event adapter

- Easy to handling low-level event
  - Listener interface ห้ามมี event ห้าม handling methods over 2 (xxxListener) and Adapter class ห้าม implement ห้าม things.
  - Each event ห้าม handling method is empty block

- If ห้าม there is adapter, define inherited class instead of implementing listener

```java
import java.awt.*;
import java.awt.event.*;
// 어댑터를 상속
class MyListener extends WindowAdapter {
    public void windowClosing(WindowEvent ev) {
        System.exit(0);
    }
}
```
**ActionEvent and ActionListener**

- **Major methods of ActionListener**
  - `void actionPerformed(ActionEvent ev)`
  - Example of event occurrence
    - Click a button
    - Click a menu
    - Press enter key at TextField
    - Double click a element of list

- **Major methods of ActionEvent**
  - `String getActionCommand()`
    - Return a command name
  - `Object getSource()`
    - Inherited method at EventObject
WindowEvent and WindowListener

- **Major methods of WindowListener**
  - Case changing the state of windows
  - void windowActivated(WindowEvent ev)
  - void windowClosed(WindowEvent ev)

- **Major methods of WindowEvent**
  - int getNewState()
    - 0 is common state
  - int getOldState()
  - Window getWindow()
ItemEvent and ItemListener

**Major methods of ItemListener**
- void itemStateChanged(ItemEvent ev)
- Select Item or release item at Checkbox, CheckboxMenuItem, Choice, List

**Major methods of ItemEvent**
- Object getItem()
- int getStateChange()
- String paramString()
AdjustmentEvent and AdjustmentListener

- **Major methods of AdjustmentListener**
  - `void adjustmentValueChanged(AdjustmentEvent ev)`
    - Case changing the state of scroll bar

- **Major methods of AdjustmentEvent**
  - `int getValue()`
  - `int getAdjustmentType()`
    - `UNIT_INCREMENT`, `UNIT_DECREMENT`,
    - `BLOCK_INCREMENT`, `BLOCK_DECREMENT`, `TRACK`
KeyEvent and KeyListener

- When event related with keystroke occurs, KeyEvent occurs

- Major methods of KeyListener
  - keyPressed(KeyEvent ev)
  - keyReleased(KeyEvent ev)
  - keyTyped(KeyEvent ev)

- Major methods of KeyEvent
  - char getKeyChar()
  - int getKeyCode()
  - int getKeyLocation()
  - static String getKeyText(int keyCode)
  - static String getKeyModifiersText(int)
MouseEvent and listener

- When event related with a mouse occurs, MouseEvent occurs

- Major methods of MouseListener
  - mouseClicked(MouseEvent ev)
  - mousePressed(MouseEvent ev)
  - mouseReleased(MouseEvent ev)

- Major methods of MouseMotionListener
  - mouseDragged(MouseEvent ev)
  - mouseMoved(MouseEvent ev)

- Major methods of MouseEvent
  - int getButton()
  - Point getPoint()
TextEvent and FocusEvent

- **Major methods of TextListener**: `textValueChanged(TextEvent ev)`
  - When texts are changed at TextArea and TextField, event occurs and listener executes.

- **Major methods of FocusListener**: `focusGained(FocusEvent ev)`
  - When component gets or misses input focus, event occurs and listener executes.
  - `focusLost(FocusEvent ev)`
Overview

- AWT Basic
- Use AWT
- AWT Event
- Specific things
Graphic Context

- Environments for Graphic tasks
  - EX) coordinates, background color, foreground color, font

- Major characteristic managed by graphic context
  - Object to be drawn: target object that graphic object will draw
  - Coordinate system: coordinate information for drawing base
  - Current clipping area: clipping information
    - Clipping area save information of area to be redrawn
  - Current color and font: information of color and font for drawing
Coordinate System

- General coordinate system in computer graphic
  - Left top is (0, 0)
  - X-axis increases to right side
  - Y-axis increases to down side

Ex)
Case of Rectangle width * height

Component

(width-1, height-1)
Layout Manager

- Manage Position
  - Manage component position attached to container

- Type of Layout
  - BorderLayout
  - BoxLayout
  - CardLayout
  - FlowLayout
  - GridBagLayout
  - GridLayout
  - SprintLayout
Role of Layout Manager

Without Layout Manager

With Layout Manager
Hot to Set Layout

1. Create layout manager
   `BorderLayout bm = new BorderLayout();`

2. Set layout manager of component
   `setLayout(bm);`

3. Add component
   `add(myButton);`

   - Arguments varies with layout manager
   - See layout class in API for more information
**BorderLayout**

- Organize layout like the picture on the left
- Site image on the specific position with argument adding component

```java
public LayoutFrame()
{
    setLayout(new BorderLayout);
    add(new Button("North"), BorderLayout.NORTH);
    add(new Button("South"), BorderLayout.SOUTH);
    add(new Button("East"), BorderLayout.EAST);
    add(new Button("West"), BorderLayout.WEST);
    add(new Button("Center"), BorderLayout.CENTER);
}
```
**Color and Font**

- **Color**
  - Use for setting graphic object color, R, G, B and Alpha
  - Major constructor
    - `Color(float r, float g, float b, float a)`
    - `Color(int r, int g, int b, int a)`

- **Font**
  - Font type and size used for printing text on the screen
    - **BOLD, ITALIC, PLAIN**
  - Major method
    - `setFont(Font font)`...
    - See API for more information
Draw Shape

- Fill in Shape: start with fill, Ex) fillArc, fillRect...
- Draw shape outline: start with draw, Ex) drawArc, drawRect...

Type of Shape

- Line: straight line
- Rect: rectangle
- 3Drect: 3-dimensional rectangle
- RoundRect: rounded rectangle
- Oval: ellipse
- Arc: an arc of circle, part of circle
- Polygon: polygon,
  Make with joining start point and endpoint
Draw Line

- Draw Line
  - `drawLine(int x1, int y1, int x2, int y2)`

- Draw rectangle
  - `drawRect(int x, int y, int width, int height)`
  - `draw3DRect(), drawRoundRect(), fillRect(...)`

- Draw ellipse and elliptical arc
  - Designate rectangle that ellipse will be (inscribed ellipse)
  - Elliptical arc is based on 3 o’clock and angle goes counter clockwise direction
Draw String

- Draw string using method such as `drawString()`
  - Provide font character with `FontMetrics`
  - Font have variable width, so letter has difference width
## Toolkit Class

- **Parent class of AWT GUI toolkit object**
  - Create various component of AWT
  - Connect to native component of system

- **Abstract Class**
  - Get object with `Toolkit.getDefaultToolkit()` or `getToolkit()` method of `Component`

- `getImage()` return image object right after call
  - Do not consider whether data is completely ready

<table>
<thead>
<tr>
<th>java.awt.Toolkit major method</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Dimension</strong></td>
</tr>
<tr>
<td><strong>int</strong></td>
</tr>
<tr>
<td><strong>Image</strong></td>
</tr>
<tr>
<td><strong>Image</strong></td>
</tr>
</tbody>
</table>
Toolkit Example

Toolkit example to get a lot of information

```java
public void test() {
    Toolkit toolkit = getToolkit();
    Dimension dim = toolkit.getScreenSize();
    toolkit.setDynamicLayout(true);
    add(new Label("운영체제 : "+System.getProperty("os.name")));
    add(new Label("화면 크기 : "+dim.getWidth()+" * "+dim.getHeight()));
    add(new Label("화면 해상도 : "+toolkit.getScreenResolution()+"DPI");
    add(new Label("가로 최대화 : "+toolkit.isFrameStateSupported(Frame.MAXIMIZED_HORIZ));
    add(new Label("세로 최대화 : "+toolkit.isFrameStateSupported(Frame.MAXIMIZED_VERT));
    add(new Label("Dynamic Layout : "+toolkit.isDynamicLayoutActive()));
}
```
Draw Image

- `drawImage(Image img, int x, int y, Color bgcolor, ImageObserver obs)`
  - Fill area to be drawn with bgcolor, and draw image at (x, y) position

- `ImageObserver`
  - `ImageObserver` object call `imageUpdate()` method to draw image after all image data is loaded because size of image resources is big

- Image extension/contraction
  - Image extension and contraction is easy with using argument of `drawImage()` method
Manage Image Resources

- **All Platforms except Java**
  - Other tasks are paused until image is loaded
  - Load all data once when creating image object

- **Java Platform (asynchronous processing)**
  - Method call for image object create return immediately while background thread load image

- **Ex) Print process with incomplete image data**
  - Excepting Java, Wait for loading all data complete
  - Print process with current image data at java platform
Print Image on the Screen

1. Get name and url of image file
2. Get Toolkit object
3. Load image
4. Prepare image(trace)
5. Print image

How to use

```java
URL url = getClass().getResource(file_name);
Toolkit toolkit = Toolkit.getDefaultToolkit();
Image img = toolkit.getImage(url);
```

**Context of constructor**

```java
URL url = new URL("http://java.sun.com/docs/books/tutorial/images/wood8.GIF");
imgExample = getToolkit().getImage(url);
```

**Paint method**

```java
public void paint(Graphics g){
    if (getToolkit().prepareImage(imgExample, -1, -1, this) == false)
    {
        g.drawString("Wait for Loading", 100, 100);
    } else {
        g.drawImage(imgExample, 10, 10, this);
    }
}
```
Create Image Button

- Make button more visual by inserting image

ImageButton.java

```java
public void paint(Graphics g) {
    if (xImage == null) super.paint(g);
    else {
        /* draw image at the center of button. */
        g.drawImage(
            xImage,
            (getWidth() - xImage.getWidth(this)) / 2,
            (getHeight() - xImage.getHeight(this)) / 2,
            this
        );
    }
}
```
Organization of Menu
How to Make Menu

1. Create menu bar to attaching menu
   ```java
   MenuBar myMenuBar = new MenuBar();
   ```

2. Create menu
   ```java
   Menu myMenu = new Menu("내 메뉴");
   ```

3. Create menu item and add in menu
   ```java
   MenuItem myMenuItem = new MenuItem("내 아이템");
   myMenu.add(myMenuItem);
   ```

4. Add menu in menu bar
   ```java
   myMenuBar.add(myMenu);
   ```

5. Set menu bar at frame
   ```java
   Frame myFrame = new Frame();
   myFrame.setMenuBar(myMenuBar);
   ```
Checkbox menu

... Menu myMenu = new Menu("내 메뉴"); CheckboxMenuItem myCheckboxMenuItem = new CheckboxMenuItem("내 체크박스메뉴"); myMenu.add(myCheckboxMenuItem);
... 

- Use CheckboxMenuItem instead of MenuItem
Sub menu

... Menu myMenu = new Menu("내 메뉴"); Menu mySubMenu = new Menu("내 서브메뉴"); MenuItem mySubMenuItem = new MenuItem("내 서브메뉴 아이템"); mySubMenu.add(mySubMenuItem); myMenu.add(mySubMenu);
...

- Use Menu instead of MenuItem
Exercise

- Download a WeeklyCalender.jar file and import it.
- Just fill the blank about layout, buttons, events.
  - file i/o and string processing will be next time
  - `saveWeekPlan()`, `loadWeekPlan()`