

## Section A: A One Page Discussion of the Abstract Window Toolkit (AWT)

The Abstract Window Toolkit, commonly referred to as the “AWT”, is the part of Java enabling programmers to implement GUI applications. Composed of a package of classes named “*java.awt*”, it supports everything associated with a Graphical User Interface, from buttons and dialog boxes to menus and event handling.

One of the problems of making programs for Graphical User Interfaces (such as Windows 95 and the Macintosh Operating System) is that different GUIs have different sets of facilities for providing a GUI. So, for example, a program written using Borland Delphi for Windows 98 will not work on an Unix machine. Java solves this problem by mapping the facilities of the AWT onto the facilities of the particular Operating System which the program is running on. A Java program is platform-independent i.e. it will run on virtually any machine.

The AWT, which has been present in all versions of Java, contains a set of classes (more than 50) which programmers derive new classes from to use in their programs. There are four main classes in the Abstract Window Toolkit, namely *the Component class, the Container class, the Graphics class, and the LayoutManager interface*. These classes contain pieces of code enabling the programmer, for example, to create a canvas to draw on. In fact, approximately half of the classes in the AWT are extensions of the *java.awt.Component* class. This is the foundation upon which the AWT is built.

One of the disadvantages of programming in a GUI environment is that the user can choose to do one of several things i.e. initiate different events. In a text or command based environment, programs traditionally follow a sequence of instructions, and the user typically has to follow what it says on screen. However, in a GUI environment, the user may click on buttons, select a menu command, or even execute another program whilst the first program is still running.

Fortunately, the Java programmer doesn't have to worry about tracking the mouse, reading the keyboard or similar low-level events. However, the Java programmer will have to worry about what will happen when the user, for example, clicks on a button. In this case, the programmer will have to write code to make sure that the action the user expects to take place when he or she clicks the button does indeed take place.

In summary, the AWT is a platform-independent windowing toolkit. It allows a programmer to create programs in Graphical User Interfaces, so commonly used today in many places such as business. It's major advantage is that no code alterations are needed when compiling the source code on different machines. However, its powerful features need proper understanding to be unleashed.

## Section B: A Listing of a Program using the AWT

```
/**
 *
 * AWT Demo - "Premier League Champions"
 *
 * Written by: Gareth Evans
 *
 * First Written: 17/Nov/98
 * Last Rewritten: 19/Nov/98
 */

import java.awt.* ;
import java.awt.event.* ;

public class Football extends Frame
    implements WindowListener, ActionListener

{ // start of class Football

    Canvas1 canvas;
    MenuItem start, quitprog, Arsenal, Rovers, ManUtd ;
    Button Ars, BBurn, ManU, quit;
    String champion;

/**
 *
 * constructor
 *
 */

    public Football(String ch)

        { // start of constructor method

            champion = ch;

// set up basic window
            setTitle("FA Premier League Champions") ;
            setBackground(Color.green) ;
            setSize(550, 350) ;
            addWindowListener(this) ;

// set up area for face
            canvas = new Canvas1(this) ;
            add("Center", canvas) ;

// set up area with buttons
            Panel p = new Panel() ;
            p.setLayout(new GridLayout(4,1)) ;
            Ars = new Button("Arsenal");
            p.add(Ars);
            Ars.addActionListener(this);
            BBurn = new Button("Blackburn");
            p.add(BBurn);
            BBurn.addActionListener(this);
            ManU = new Button("Manchester Utd");
            p.add(ManU);
            ManU.addActionListener(this);
            quit = new Button("Exit") ;
            p.add(quit) ;
            quit.addActionListener(this) ;
            add("East", p) ;
```

(Comments)

Import AWT

Menu Items  
Buttons  
The 'champion'  
variable

Window Title

Initial window size

Set up a panel with  
four buttons on the  
right ('East'), one  
button for each of  
the teams and one  
button to exit the  
program.

```

// set up menu system
Menu menu = new Menu("File") ;
Arsenal = new MenuItem("Arsenal") ;
menu.add(Arsenal) ;
    Arsenal.addActionListener(this) ;
Rovers = new MenuItem("Blackburn Rovers") ;
menu.add(Rovers) ;
    Rovers.addActionListener(this) ;
ManUtd = new MenuItem("Manchester United") ;
menu.add(ManUtd) ;
    ManUtd.addActionListener(this) ;
menu.addSeparator() ;
quitprog = new MenuItem("Exit") ;
    menu.add(quitprog) ;
    quitprog.addActionListener(this) ;
MenuBar menuBar = new MenuBar() ;
menuBar.add(menu) ;
setMenuBar(menuBar) ;

} // end of constructor method

/**
 *
 * main
 *
 */

public static void main(String[] args)

    { // start of main method

        Football f;
        if (args.length == 0)
            f = new Football("Champions");
        else
            f = new Football(args[0]);

        f.setVisible(true) ;

    } // end of main method

/**
 *
 * actionPerformed ocde
 *
 */

public void actionPerformed(ActionEvent event)

    { // start of method actionPerformed

        // deal with "Quit" commands
        if ((event.getSource() == quit) |
            (event.getSource() == quitprog))
            {
                dispose();
                System.exit(0);
            }

        // deal with "Arsenal" commands
        else if ((event.getSource() == Ars) |
                (event.getSource() == Arsenal))
            {
                champion = "Arsenal";
                canvas.repaint() ;
            }
    }

```

Set up a menu system, with items in a 'File' menu as follows: one menu item for each of the teams, and then an exit item after a menu separator.

The main method invokes the window and makes it visible on the screen.

The 'If' statement here checks to see if the source of the action was from the clicking of the 'exit' button *or* from selecting the 'exit' command from the menu system. If this is true, then the program terminates.

Here, the 'If' statement checks for 'Arsenal' related actions and repaints the canvas (after changing the variable 'champion').

```

        // deal with "Blackburn" commands
        else if ((event.getSource() == BBurn) |
                (event.getSource() == Rovers))
        {
            champion = "Blackburn";
            canvas.repaint() ;
        }

        // deal with "Manchester United" commands
        else if ((event.getSource() == ManU) |
                (event.getSource() == ManUtd))
        {
            champion = "Manchester United";
            canvas.repaint() ;
        }
    } // end of method actionPerformed

public void windowClosing(WindowEvent event)

    { // start of method windowClosing

        dispose();
        System.exit(0);

    } // end of method windowClosing

public void windowOpened(WindowEvent event) {}
public void windowIconified(WindowEvent event) {}
public void windowDeiconified(WindowEvent event) {}
public void windowClosed(WindowEvent event) {}
public void windowActivated(WindowEvent event) {}
public void windowDeactivated(WindowEvent event) {}

} // end of class Football

class Canvas1 extends Canvas

    { // start of class Canvas1

        Football parent ;

    /**
     *
     * constructor
     *
     */

        public Canvas1(Football f)

            { // start of constructor method

                parent = f;

            } // end of constructor method

    /**
     *
     * paint
     *
     */

        public void paint(Graphics g)

            { // start of paint method

```

The code for the 'Blackburn' and 'Manchester United' sections shown here is identical to the 'Arsenal' code on the previous page, except of course that the code deals with the events relating to the correct team and changes the variable 'champion' accordingly.

If the window is closed for any reason, this code makes sure we dispose of the resources used and closes the window.

```

// set up some dimensions for the drawing
Dimension d1 = getSize() ;
int cx = d1.width / 2,
cy = d1.height / 2,
centrecircleRadius = 50,
arcRadius = 5 ;

// draw the frame
g.setColor(Color.white) ;
g.drawRect(10, 10, d1.width - 20, d1.height - 20) ;

// draw the pitch
g.setColor(Color.white) ;
g.drawRect(10, d1.height/2-60, 80, 120);
g.drawRect(d1.width - 90, d1.height/2-60, 80, 120);
// 18 yard boxes
g.drawRect(10, d1.height/2-30, 30, 60);
g.drawRect(d1.width - 40, d1.height/2-30, 30, 60);
// 6 yard boxes
g.drawLine(d1.width/2, 10, d1.width/2,
           d1.height - 10); // centre line
g.drawOval(cx - centrecircleRadius,
           cy - centrecircleRadius,
           centrecircleRadius * 2,
           centrecircleRadius * 2); // centre circle
g.drawArc(70, d1.height/2-40, 40, 80, -90, 180);
g.drawArc(d1.width-110, d1.height/2-40,
           40, 80, 270, -180); // "D" arcs

// set up screen Fonts
Font f1 = new Font("TimesRoman", Font.PLAIN, 14) ;
Font f2 = new Font("TimesRoman", Font.BOLD, 14) ;
FontMetrics fm1 = g.getFontMetrics(f1) ;
FontMetrics fm2 = g.getFontMetrics(f2) ;
g.setColor(Color.black);

// display graphics
String filenameA = "Prem.gif";
Image imageA =
    Toolkit.getDefaultToolkit().getImage(filenameA);
g.drawImage(imageA, d1.width-145, 11, this);

if (parent.champion == "Arsenal")
    { // start of Arsenal Section

        // Show Arsenal's text
        String a1 = "Arsenal " ;
        String a2 = "1997-1998" ;
        int aw1 = fm1.stringWidth(a1) ;
        int aw2 = fm2.stringWidth(a2) ;
        g.setFont(f1) ;
        g.drawString(a1, 20, d1.height-15) ;
        g.setFont(f2) ;
        g.drawString(a2, d1.width - 80,
                    d1.height-15) ;

        // Display Arsenal glyph
        String filename1 = "Arsenal.gif";
        Image image1 =
            Toolkit.getDefaultToolkit().getImage(filename1);
        g.drawImage(image1, 15, d1.height - 70,
                    this);

    } // end of Arsenal Section

```

Here we set up variables to make sure we draw the objects in the correct places at all times.

Draw the 'master' frame.

This code draws the markings of a football pitch on screen. It uses commands such as 'drawRect' to draw the pitch perimeter and the penalty area and 'drawArc' to draw the 'D' arcs. It uses the variables 'd1' and 'd2' to place the components correctly.

Here we specify which font we want to use and get the dimensions of the font.

Using the command 'drawImage', we display a GIF graphic on screen.

If the variable 'champion' contains the text 'Arsenal', then we execute the code shown on the left.

Here we initialise two Strings and display them on screen.

Here we display an image, again using the drawImage command.

```

else if (parent.champion == "Blackburn")
    { // start of Blackburn Section

        // Show Blackburn's text
        String b1 = "Blackburn Rovers " ;
        String b2 = "1994-1995" ;
        int bw1 = fm1.stringWidth(b1) ;
        int bw2 = fm2.stringWidth(b2) ;
        g.setFont(f1) ;
        g.drawString(b1, 20, d1.height-15) ;
        g.setFont(f2) ;
        g.drawString(b2, d1.width -80,
                    d1.height-15) ;

        // Display Blackburn glyph
        String filename2 = "Blackburn.gif";
        Image image2 =
Toolkit.getDefaultToolkit().getImage(filename2);
        g.drawImage(image2, 15, d1.height - 70,
                    this);

    } // end of Blackburn Section

else if (parent.champion == "Manchester United")
    { // start of Manchester Untited Section

        // Show Manchester United's text
        String m1 = "Manchester United " ;
        String m2 = "1992-1993" ;
        String m3 = "1993-1994" ;
        String m4 = "1995-1996" ;
        String m5 = "1996-1997" ;
        int mw1 = fm1.stringWidth(m1) ;
        int mw2 = fm2.stringWidth(m2) ;
        g.setFont(f1) ;
        g.drawString(m1, 20, d1.height-15) ;
        g.setFont(f2) ;
        g.drawString(m2, d1.width -80,
                    d1.height-75) ;
        g.drawString(m3, d1.width -80,
                    d1.height-55) ;
        g.drawString(m4, d1.width -80,
                    d1.height-35) ;
        g.drawString(m5, d1.width -80,
                    d1.height-15) ;

        // Display Manchester United glyph
        String filename3 = "Man Utd.gif";
        Image image3 =
Toolkit.getDefaultToolkit().getImage(filename3);
        g.drawImage(image3, 15, d1.height - 70,
                    this);

    } // end of Manchester United Section

else
    { }

```

Here we execute similar code as we saw on the previous page to deal with the instance that the 'champion' variable contains the string 'Blackburn'.

Note: The image, which here is 'Blackburn.gif', must be stored in the same directory as the source code for it to be displayed.

The Manchester United section is bigger because there is more information to be displayed about Manchester United. We therefore just have to use more strings to display the information we want to display.

```

// write the general text
String s1 = "FA Premier League " ;
String s2 = "Champions" ;
int w1 = fm1.stringWidth(s1) ;
int w2 = fm2.stringWidth(s2) ;
g.setFont(f1) ;
g.drawString(s1, 20, 30) ;
g.setFont(f2) ;
g.drawString(s2, 20 + w1, 30) ;

} // end of method paint

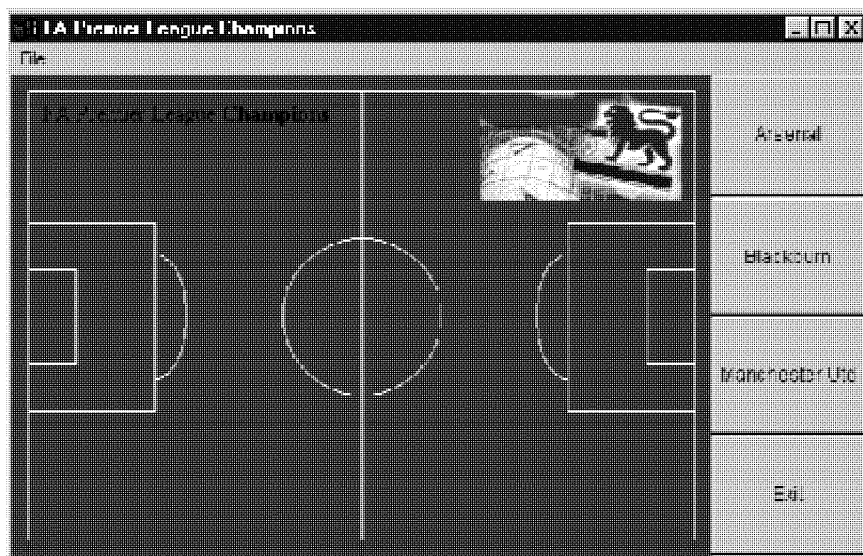
} // end of class Canvas1

```

Here we display on screen text that is shown no matter what event has occurred (even if no event has occurred).

End of program!

The purpose of this program is to allow the user to get information regarding the winners of the F.A. Carling Premiership from the first champions of the Premiership (1992/1993) to the present champions (1997/1998 season). When the program starts, it displays a football pitch and four buttons, together with a menu system. Using these components, shown below, the user is allowed to select whether he or she wants to exit the program, or either click on the buttons or use the menu system to see when the teams shown won the F.A. Carling Premiership.



If the user wants to exit the program, then he or she simply clicks on the exit button or uses the menu system. The program then shuts down as expected. However, if the user wants to see information regarding the winners of the Premiership, he or she would click on one of the other buttons or select another item from the menu to see the information shown below. When the user clicks on a team name, then (1) a graphic is shown representing the kit colours of the team, and (2) the years in which that particular team won the Premiership are also shown.

