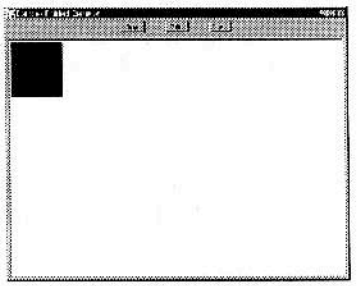
CHAPTER 3 : USING SIMPLE GRAPHICS

In this Chapter...

You will learn about:
the setColor, fillRect, fillOval, fillStar methods, Color col = new Color (r,g,b);

You will learn how to:
draw rectangles, circles and stars, use find and replace, use built-in help, create new Color objects

3.1 Graphics output on the Console window



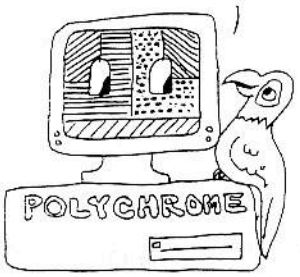
- Start a new HSA Console Application called **BlueRect**
- Add your name and today's date as a comment statement in line 2.
- Type in the following program code after the `c = new Console ();` statement:

```
c.setColor (Color.blue);
c.fillRect (0, 0, 100, 100);
```

Be careful to copy these lines exactly. The dots and the brackets are important, and so is the capitalisation.

Note the American spelling for Color.

The above lines of code
- use the *setColor* method of the *Console* class to set the drawing color on the *Console* window object called "c" to blue
- draw a filled rectangle with upper-left corner of (0, 0) and a width and height of 100 pixels (dots on the screen).



- Save and Run the program, checking that you do get a blue rectangle in the top left corner of the screen.
- Make the following changes to the code in your BlueRect program. Save each change, then Run the program. Write down what effect the change has on the output each time.

3.1.1 Change `c.fillRect (0, 0, 100, 100);` to `c.fillRect (80, 20, 100, 100);`

- Click Save, then Run. Write down what difference this makes:

.....

3.1.2 Now change it to `c.fillRect (80, 200, 100, 100);`

- Click Save, then Run. Write down what difference this makes:

.....

3.1.3 Now change it to `c.fillRect (80, 200, 300, 100);`

- Click Save, then Run. Write down what difference this makes:

.....

3.1.4 Now change it to c.fillRect (80, 200, 150, 350);

▶ Click Save, then Run. Write down what difference this makes:

.....

3.1.5 Change the last 2 figures back to 100, 100; and also change the first 2 figures so that the blue rectangle is in the top right corner of the output screen. Write down the figures that you used in the c.fillRect command to make this work:

`c.fillRect (`.....

3.1.6 Change the last 2 figures to 5, 5 so that you will have a very small blue rectangle; and also change the first 2 figures so that this appears in the top right corner of the output screen. Write down the figures that you used:

`c.fillRect (`.....

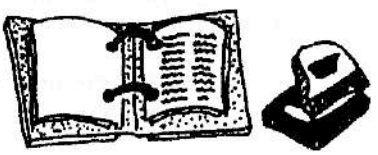
3.1.7 Add code so that another small red rectangle (size 5 x 5) is drawn in the bottom left corner of the output screen. Write down the figures that you used for this rectangle:

`c.fillRect (`.....

3.1.8 Use your answers from above to work out the approximate dimensions in pixels of the Console output window by creating a rectangle to fill the whole window:
The console window has the dimensions:
Across: ..... Down: .....

3.1.9 Add code to draw 3 more small rectangles (all size 5 x 5) as follows:

- a green one in the top left corner
- a yellow one in the bottom right corner
- a pink one in the middle of the screen (approximately).



▶ Save and Run program. Correct it until it works perfectly!

▶ Print your program if you have a printer available and file it under Chapter 3.

The format of `fillrect` is:

`c.fillrect(x,y,width,height);`

`x` and `y` are the starting coordinates of the rectangle from the top left hand corner.

`width` and `height` are the dimensions of the rectangle. i.e. across and down.

Java follows the math conventions of coordinate axes by moving **across** first and then **down**.

3.2 Using Replace to change all occurrences of a word in a program

Sometimes it is useful to be able to change the class name, or other words that you have used in your program.

Here you will use **Replace** to change the name of this class from *BlueRect* to *Rectangles*, because the name *BlueRect* is no longer appropriate for this program.

▶ Start *Ready* and open the *BlueRect* program which you saved onto your disk.

- ▶ Put the cursor at the very beginning of the program, then use **Replace** by
  - pressing the **Replace** button in the button bar, or
  - selecting **Replace** from the **Search** menu.

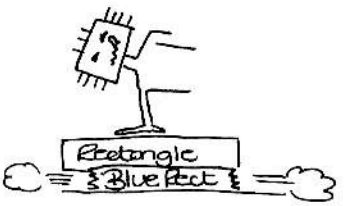
The Replace dialog box should appear.

- ▶ Enter "BlueRect" in the *Find What* text box and "Rectangles" in the *Replace With* text box.

"BlueRect" is called the Search string and "Rectangles" is the Replace text.

- ▶ Click the *Find Next* button.

The selection jumps to the word "BlueRect" in the class on the line:



`// The "BlueRect" class` and highlights it.

- ▶ Click the *Replace* button in the dialog box.

"BlueRect" is replaced with "Rectangles" and the editor jumps to the next occurrence of "BlueRect", which is in the line:

```
public class BlueRect
```

- ▶ Continue to replace all the occurrences of "BlueRect" with "Rectangles".

When there are no more occurrences, *Ready* notifies you that it could not find the search string.

Now there is a problem with the file name, *BlueRect,* as the name of the main class cannot be different than the file name, so the filename must be changed.

- ▶ From the File menu, choose **Save As**

- ▶ Type the new file name `Rectangles` to correspond with the new class name, then choose the correct folder or disk and click OK to save the program with the new name.

3.3 Using the built-in help

Open the Java Documentation, either by using:

- the Help menu in **Ready to Program Help** | Help with Java class libraries

  or
- Windows Explorer | Program Files | Ready to Program | Support | Help | Java11Doc.chm

The left frame enables you to find what you are looking for, and the right frame gives the explanation.

Until you get more used to it, this Help is quite hard to use!

3.3.1 Information about a rectangle

- ▶ Make sure that the Index tab in the left frame is selected.

- ▶ Type in the keyword *rectangle*. (As you type, the list of items changes.)

- ▶ Click the Display button near the bottom of the left frame. The right frame should now display information about a rectangle.

- Complete the first full sentence under **Class java.awt.Rectangle** | public class **Rectangle**:

A rectangle specifies an area .....

.....

.....

When you used the *fillRect* method to draw a colored-in rectangle in the previous program, there were 4 numbers in brackets after this command. Eg: `fillRect (20, 10, 200, 100);`

These 4 numbers in brackets are called parameters.

- Write down what each of these 4 numbers represents in terms of the sentence you just copied down:

.....

.....

.....

.....

- Read on in the Help to see what happens if a rectangle is given a negative value for the width or height. Write down a summary of what is said: .....

.....

.....

.....

.....

3.3.2 Information about the color command

- Type in the keyword *color* then click the Display button.
- Scroll down the right frame until you can see all possible colors that can be used.
- Make a list of all the colors that can be used by their color name:

.....

.....

.....

- Minimize the Java 1.1 Doc window, so that it is shown in the Taskbar at the bottom of the screen.

3.4 Creating your own colors

As well as the colors given in the list you copied down in section 3.3.2, you can make up your own colors.

- Open the `Rectangles` program, if you do not have it on the screen.
- Write down the last setColor command used in the program: .....
- This command was used to draw a rectangle of what color? .....

- Replace this line with the following 2 lines:

```
Color col = new Color(255, 0, 255);
c.setColor(col);
```

A new Color object called **col** is created, which uses the values (255,0,255) to give it its color.
The color is set to this new color.
This will be explained in more detail later.

- Click Save, then Run.

- Write down what change occurred to the pink rectangle in the middle of the screen, as a result of these 2 new lines: .....

- How many **parameters** does **Color** have? .....

Parameters are the figures in brackets after the name.

The three parameters of Color give the quantity (in the range 0 to 255) of each of the colors Red, Green, Blue that is used to make up a color.

So Color(255, 0, 255) means 255 of Red and 255 of Blue, but no Green, which results in the color Purple.

EXERCISE 3A

1 Start a new HSA Application called `MyColors` for this exercise. You will create various different color combinations to fill in a rectangle.

- Add a comment with your name and today's date.

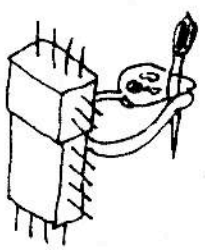
- Type in the following lines to set the starting color:

```
Color col = new Color(255, 0, 255);
c.setColor(col);
```

- Type a statement to draw a fairly large filled-in rectangle near the middle of the screen so that you will clearly be able to see the color changes that take place.

- Save, then Run. Write down what color rectangle results ..... *(It should be Purple.)*

- Change the 3 parameters of Color as shown in the table on the next page to create new colors. Each time, Save then Run. Write down what color rectangle results.



`Color( R, G, B );`	**Color produced**
`Color( 0, 0, 0 );`	
`Color( 200, 200, 200 );`	
`Color( 255, 255, 255 );`	
`Color( 255, 0, 0 );`	
`Color( 0, 255, 0 );`	
`Color( 0, 0, 255 );`	
`Color( 160, 100, 0 );`	
`Color( 0, 200, 200 );`	
Try to make up some of your own colors. Write down the interesting ones.	

2 Use the **HSA Console Application Boilerplate** to start a new class called **Circles**.

▶ Add a comment with your name and today's date.

▶ Save on disk with the name `Circles`

▶ Add code to draw a large (250 x 250) colored-in magenta circle in the approximate centre of the screen.

Hint: You cannot use the fillRect command. Why not? .....

Try a similar command for circles. Write down what you tried: .....

.....

▶ If you cannot find a command that works, use the built-in Help (Java 1.1. Documentation) as described below.)

- Load the Java 1.1. Doc (as described previously).
- In the RIGHT-HAND-SIDE frame, click on the Index hyperlink at the top right of the screen, then click on the letter F.
- Scroll down until you get to "fill". Find the *fillRect* command.

▶ Write down what is in brackets after *fillRect* .....

In Java, **int** means that the "data type" of each one is integer, so *fillRect(int, int, int, int)* means that the *fillRect* command requires 4 parameters and these must all be integers.

▶ How many parameters (values in brackets) does *fillRect* need? .....

▶ Look at the rest of the "*fill...*" commands in the Help. See if you can find one that will draw a circle.

▶ Go back to your Circles program and try to make it work.

- When your program does draw a colored-in circle, write down
  - the command you used .....
  - the complete code you added: .....
  .....
- How many parameters has the *fillOval* command? .....
- What data type are these parameters? .....
- Change the command you used to *drawOval* then Save and Run.
- Write down the difference this makes .....
  .....

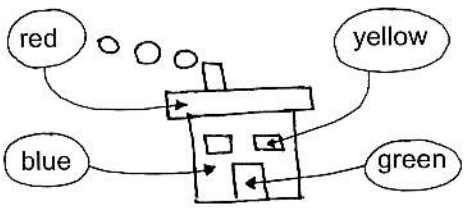
EXERCISE 3B

- Code Java Applications using the HSA Console Application Boilerplate for each of the following.

  - Use the class name given next to the number of the exercise.
  - Save each program on disk with the same name as the class.
  - Put your name and today's date in each program.

1 **House**

Draw a picture of a house,
using the colors indicated.
Make up the sizes for each rectangle.



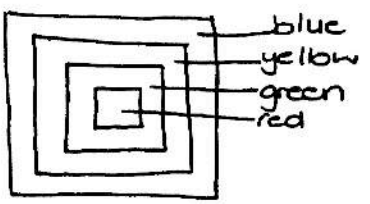
2 **Moon**

Fill the output screen with black, then draw a yellow circle, and on top of it, but slightly higher and to the right, draw a black circle which overlaps the yellow circle, to make it look like a crescent moon.

Use the *fillStar* method to position colored-in stars around the moon.

3 **ColSquares**

Display the following, with each square being 100 pixels smaller than the one below it. Use the colors shown.



KWIKWIZ

1. Explain in your own words what the following methods do:

   `setColor` `fillRect` `drawRect`

   .....
   .....
   .....

2. What is a parameter? .....
   .....

3. In the statement `c.setColor (Color.blue);`

   a. which is the **object** and which is the **method** in `c.setColor` ? .....
   .....
   which is the **method** in `Color.blue` ? .....
   .....

4. Explain in your own words what the following section of code does:

```
Color col = new Color (255,255,255);
c.setColor (col);
```

   .....
   .....
   .....
   .....

5. Why does the following section of code result in only 1 red rectangle being displayed? What happened to the blue rectangle?

```
c.setColor (Color.blue);
c.fillRect( 0,0,100,200);
c.setColor (Color.red);
c.fillRect( 0,0,100,200);
```

   .....
   .....
   .....
   .....

6. Write down a section of code which will result in the entire output Console window being colored in black: .....
   .....

7. Write down the name of the **method** that can be used to draw the outline of a **circle**:
   .....