Colour Concepts

How Colours are Displayed

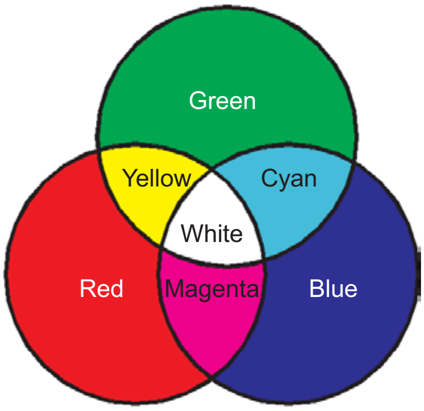
Have you looked closely at your television screen recently? It's in full colour, showing every colour and shade that your eye is capable of seeing. And yet your television can generate only 3 basic colours: red, green and blue.

Every other colour that you see on the screen is made up from those 3 colours. For example, to show the colour yellow, the television combines the colours red and green; red, green and blue together produce white; when none of the basic colours are generated, we have black.

This is known as the **additive colour process** and the colours red, green and blue are known as the **primary colours**. The basic colours that can be constructed from the 3 primary colours are shown in FIG-5.1.

FIG-5.1

The Additive Colours System



As you can see from the figure above, green and blue combine to give a colour called cyan while red and blue give magenta.

To create other colours and shades we only need to vary the brightness of the primary colours. Hence, to create orange we have an intense red, a less intense green, and no blue.

In a computer system the colour of any spot on the screen is recorded as a series of 3 numbers. The numbers represent the intensities of the red, green and blue components (in that order) that make up the colour of that spot. Each number can range between 0 and 255; 0 means that that colour is not used while 255 means that the colour is at full strength. Hence, if a spot on the screen shows brilliant yellow that would be recorded as 255, 255, 0. This means that red is at full intensity, green is at full intensity and that blue is switched off.

How Colours are Recorded

There are two basic ways to record colour information. One method is to do the obvious and record the red, green and blue (RGB) intensities for each part of the image. Let's assume that an image has a yellow spot followed by a cyan spot. This would be recorded as

```
255, 255, 0    0, 255, 255
```

When an image contains only a limited number of colours, as might be the case for example in a sunset, then an alternative approach is to use a **colour palette**. A colour palette is a table containing 2 columns of information. The right-hand column of the table contains the RGB values for the colours used in the image. For example, if an image used only the colours yellow, white, cyan and black, the right hand column of the palette would be as shown in FIG-5.2.

FIG-5.2

The Colour Palette (1)

The order in which the colours are listed is immaterial.

	Colour
	255, 255, 0 255, 255, 255 0, 255, 255 0, 0, 0

Yellow
White
Cyan
Black

In the first column of the table we enter a code for each of the colours, starting at zero and incrementing by one for each subsequent row (see FIG-5.3).

FIG-5.3

The Colour Palette (2)

Code	Colour
0	255, 255, 0
1	255, 255, 255
2	0, 255, 255
3	0, 0, 0

Yellow
White
Cyan
Black

Now, if an image contains a yellow spot followed by a cyan spot we can record this using the code value for each of these colours:

```
0  2
```

Since different images will use different colours and hence require different colour palettes, the palette itself must be recorded as part of the image file.

Activity 5.1

Using paper and pencil, construct the palette for an image using the colours red, green, blue, white, black, cyan, magenta and yellow.

Image Types

Introduction

In a computer system there are two fundamentally different ways of recording images. These are:

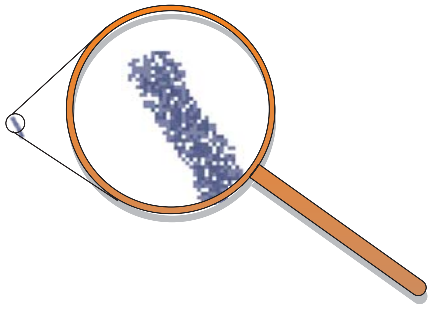
- bitmap
- vector

Bitmap Images

In a bitmap image the picture is made up of tiny coloured rectangles known as **pixels**. These pixels are so small that you're unlikely to notice them, but if you magnify the image, then the pixels from which it is constructed become obvious. FIG-5.4 shows the effect of magnifying a section of an image.

FIG-5.4

Bitmaps and Pixels



This is the type of image created when using a scanner or a digital camera.

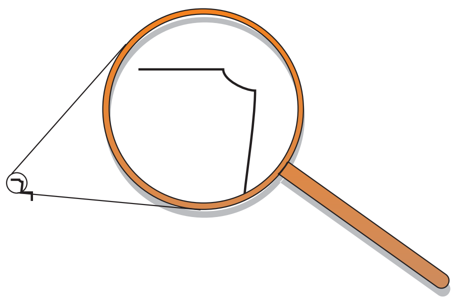
A bitmap image file holds information about the amount of red, green and blue in each pixel of the image.

Vector Images

A vector image contains only simple lines and shapes. Details of these lines and shapes are held in the file as a set of mathematical formulae. For example, a circle in an image could be recorded by storing the position of the centre of the circle, its radius and the colour and thickness of its circumference. When the image is to be displayed, the software interprets these formulae to reconstruct the original picture. This approach results in a perfect representation of the image, no matter how much you magnify the picture (see FIG-5.5).

FIG-5.5

Vectors



Almost every image that you see on a standard Internet page will be a bitmap image although vector images are used in more specialised situations.

File Formats

However, this isn't the end of the story. Bitmap images can be stored in many file formats. Probably the simplest of these is Microsoft's Bitmap format. Files using this structure have the extension *.bmp*. In Microsoft Bitmap files an exact copy of the original image is saved. Three numbers, each between 0 and 255, represent the intensity of the red, green and blue in each pixel. This means that an image which is 200 pixels wide by 150 pixels high needs

A **byte** is the amount of computer memory required to store any number between 0 and 255.

1kb = 1024 bytes

200 x 150 x 3 bytes of storage

that is

90,000 bytes   or   88kb

Given that the text on a web page may require only 2kb or 3kb of storage we're making a large increase in a page's size by including even a small image.

Luckily, however, there are ways of reducing the number of bytes needed to store an image. Such an image is said to be **compressed**.

One method of compressing an image is to recognise when an image contains a group of contiguous pixels of identical colour and save that information in a new way. For example, in a Microsoft bitmap image, if we had three yellow pixels in a row we would record that using the values

```
255 255 0   255 255 0   255 255 0
```

As you can see this requires 9 bytes of storage.

But an alternative way of recording the same information is to define how many contiguous pixels are in the same colour and the RGB values of that colour:

```
3 255 255 0
```

Now only 4 bytes are need to record the information.

As long as there are large blocks of identical colour in an image this method can reduce the size of a file considerably.

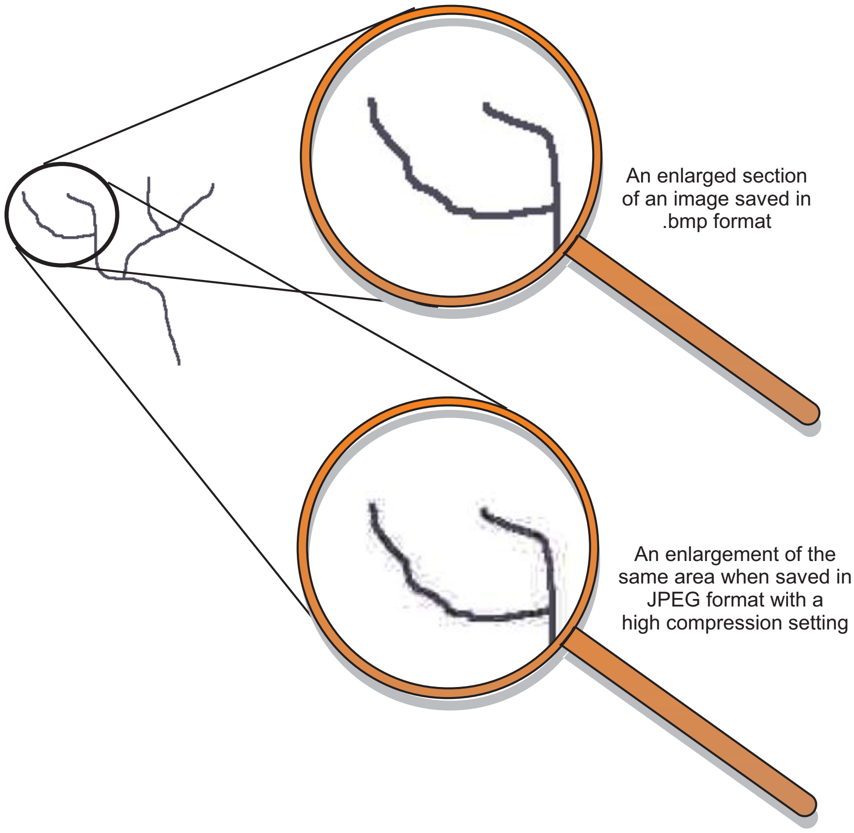
This type of compression is known as **run-length encoding**.

A second compression method is known as **JPEG compression**. This is a much more mathematically based compression technique, an explanation of which is outside the scope of this text. However, we can summarise its effects by saying that it can create files of around 5% of the size of a *.bmp* file containing the same image. The amount of compression can be set when saving the file. Files of this format have the extension *.jpg* or, more rarely, *.jpeg*.

However, the excellent compression ratio achieved by the JPEG format comes at a price. JPEG is a **lossy** compression technique. That means that when you store an image in JPEG format and then display it again, it will have lost some of the detail that was in the original uncompressed image. The more the image is compressed the more detail is lost. Even worse, the compression introduces random pixels of colour (known as **artifacts**) into the image (noticeable near the edges of shapes). An example of the artifacts caused by the JPEG compression technique is shown in FIG-5.6.

FIG-5.6

Artifacts



Images for the Web

In the early stages of the World Wide Web, text was all that could be placed on a page. Images involved too large a data transfer to be practical. However, as transmission rates increased and methods of compressing image files were developed, the inclusion of images on a web page became almost a necessity.

When an image is used on a web page it is almost always either a GIF file (which uses run-length encoding and a colour palette) or a JPEG file. Which of these two formats are used depends on the type of image involved. We may summerise the formats by stating the characteristics of each:

GIF Files

- Maximum of 256 colours
- Good for diagrams and drawings where there are large blocks of colour
- The detail of the original images is retained
- One colour can be set as transparent
- Can be used to create simple animation

JPG Files

- Full colour (16.7 million colours)
- Good for photographs
- Some detail can be lost when using this format
- The size of the file can be varied when saving
- High compression can add artifacts to the image

The Image Tag <img />

As we've seen, text displayed on a web page is included in the XHTML file. However, images shown on the page are stored in separate files. The XHTML file uses the `<img />` tag to position an image and identify the file in which it is held.

The `<img />` tag is positioned in the XHTML file corresponding to the point where the image is required on the web page.

Like a few other tags, the `<img />` tag is not paired.

The *src* Attribute

The name of the image file is given in the *src* attribute (short for *source)*. For example, we can include the file *face.jpg* within a web page using the tag:

```
<img src="face.jpg" />
```

This assumes that the file, *face.jpg*, is held in the same directory as the XHTML file containing the `<img />` tag. If the image is stored elsewhere, then path information will be required within the *src* attribute as in:

```
<img src="C:/images/face.jpg" />
```

You can find all of the images used in this chapter on our web site -

www.digital-skills.co.uk

Activity 5.2

Create a new XHTML file called *images.html* containing the following code:

```
<html>
    <head>
        <title> Using Images </title>
    </head>
    <body>
        <h1 align="center"> Including Images </h1>
        <img src="ailsa.jpg"/>
    </body>
</html>
```

Copy the file *ailsa.jpg* into your directory.

View *images.html*.

The *width* and *height* Attributes

We can specify the dimensions of an image (in pixels) within the `<img />` tag using the *width* and *height* attributes.

Although these are never required by the browser, adding them has two advantages:

- The browser can construct the basic layout of the page before the image

  has been downloaded. This allows text, which usually arrives first, to be displayed at the correct position right from the start rather than forcing an adjustment in the layout once the image arrives.

- By making the *width* and *height* values within the tag greater than the actual dimensions of the image we can magnify the image without