Introduction

A little bit of history…

- **HTML 2.0** was the first "standard" form of HTML, released in 1995 (and yes, that's also how long I've been writing Web sites!)
- **HTML 3.2** became the standard in 1997. One of the biggest changes was the introduction of **tables**; HTML 2.0 did not allow Website authors to use tables.
- Up to now we've been using HTML 3.2 because it works, and because it's a nice and easy place to start.
- At the end of 1996 a new feature was added to Web sites, which became more widely used from around 2000: **Cascading Style Sheets** or **CSS** for short.
- Just to complete the history, HTML 4.01 came out in 1999 (that's when `<font>` was officially dropped!), and was the standard until 2014, when HTML 5 was published. The latest standard (in 2020) is HTML 5.2.
- As you have seen, HTML 3.2, which we have been using so far, still works – for now.

Why do we need to use CSS, and how does it help us?

Like I just said, HTML 3.2 still works – for now. The truth is that some of the ways we've been changing things like font and using the `<font>` … `</font>` tags are a little bit "old fashioned" even if they're nice and easy to use. So, there are a few reasons we need to start using **cascading style sheets** (**CSS**):

- First, we need to use CSS because it makes sure our Web sites are likely to work for longer, when new Web browsers are released.
- Second, CSS allows us to define our own **custom styles**. For example to choose the exact font size, colour, alignment … we want to use every time we use `<h1>` … `</h1>` – or any other header format – or several other formats…
- Third, we can then use the styles we have defined, on every page of our Web site, without having to type the whole lot every time – we can specify in our style sheet "this is how I want `h1` to look every time I use it".

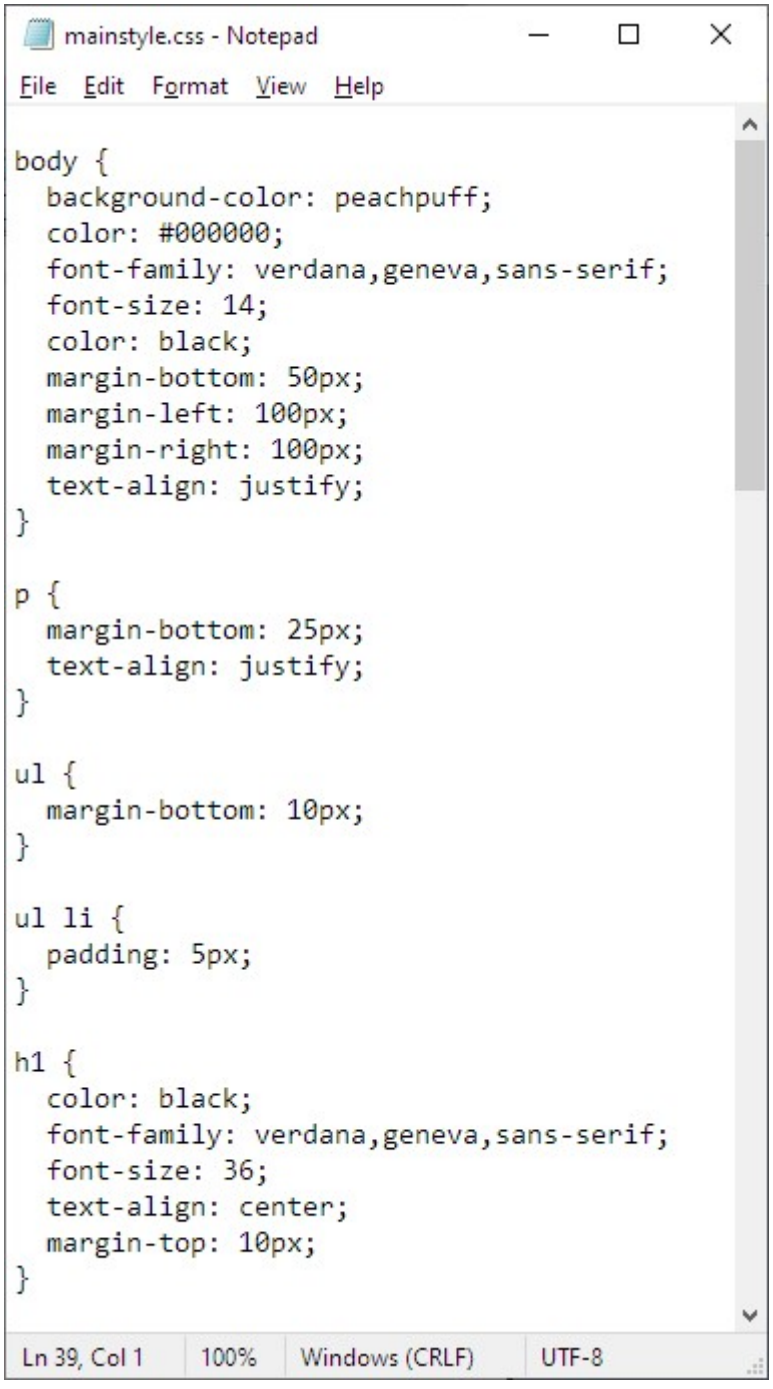
OK, so how does it work?

You can choose the style for you headers `<h1>` through to `<h6>`, your paragraphs, your menu buttons, and loads more. You can then write a style sheet with this information on it (in CSS code). Once you've written and saved your style sheet, you then just have to link to it on each of the pages of your Web site, for those styles to work all the way through that page.

If you have a big site, you can have more than one style sheet. For example, if you visit www.resources4learning.org you will see many of the pages have a salmon-coloured background, and that all the `<h1>` headers are centred. If you go into the Primary section, you'll see the style of those pages is different (there's a different background colour for a start …) – they use a different style sheet, and that style sheet can cascade (repeat) across all the pages in that section. That's why they're called Cascading Style Sheets or CSS.

What does a CSS look like?

Here's a screen shot of part of the main cascading style sheet for the resources4learning.org Web site:



At the top, you'll see, just as we save html files with the `.htm` suffix to the filename (remember that from Lesson 1?) so we need to save our style sheets with the `.css` suffix.

Next, you'll see a chunk of code telling the browser how we want each **element** of our web site to look and behave.

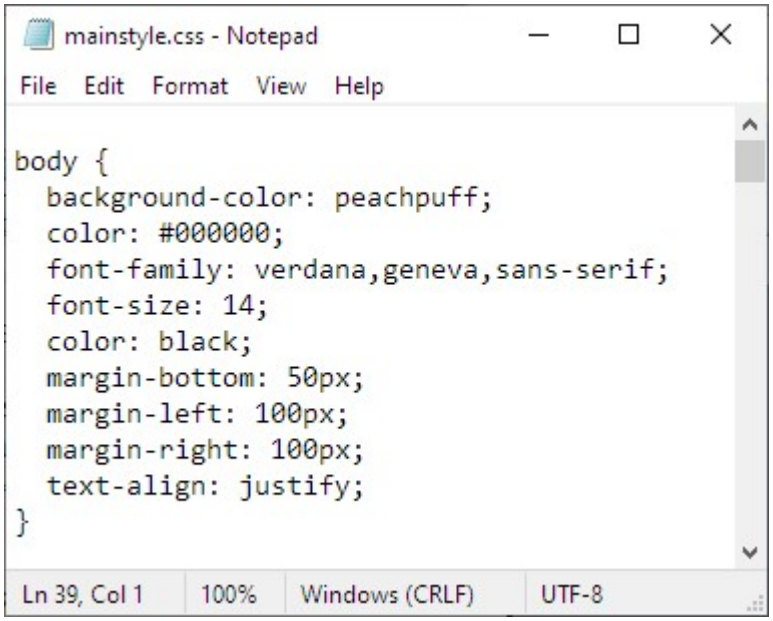
The first element of the page we need to define is the `<body>` and you'll see I've chosen a list of **attributes** that I want to apply to the body of every page in my Web site.

You'll see I've gone on to tell the browser some attribute I want to apply to other elements – for example, I've told it always to leave 25 pixels of space after each paragraph; 10 pixels after each unordered list; 5 pixels after each list item; and 10 pixels above each `<h1>` header. I've also used a new way to tell it to automatically `center` every `<h1>` header.

Wow! That's a lot to take in from just one page. Let's break down each of those chunks of code, and see what they do. Once you get the hang of this stuff, you'll see how powerful it is, and how much time and typing it can save you, when your Web site has more pages. It's also really useful to have a good grip on this stuff when you get to start GCSE Computer Science.

CSS – how the code works

Let's take a look at the chunk of css code I've used to define how the body of my pages will look. Here's that section of the code again:



Look at the first line. We start by telling our css which **element** of the page we're going to describe.

Here we're telling it about the `body` of the page.

Next, just like the < and > brackets we use in HTML tags to start and end a tag, we need to show where the **attributes** for each element begin and end.

In CSS we use the curly { and } brackets.

Once we've said which element we want to define, and we've put our curly brackets in, we can write our **definition** for that element inside those curly brackets. The definition is the set of instructions we write to say how that element of the page will look, for any page on our Web site.

The definition for `body` on this page uses many attributes:

`background-color: peachpuff;`	This one's fairly easy to understand.
`color: #000000;`	Here I'm saying I want all the text in black. #000000; is another way to describe black.
`font-family: verdana,geneva,sans-serif;`	Now I have to describe the kind of font I want the reader to see. CSS names a font family, since we don't know what fonts will be available on each reader's browser.
`font-size: 14;`	This one's fairly easy to understand.
`color: black;`	This is the font colour
`margin-bottom: 50px;`	These attributes tell the browser how much margin I want at the bottom and to each side of my page – just like the margins you use when you write something.
`margin-left: 100px;`	
`margin-right: 100px;`	
`text-align: justify;`	This one's fairly easy to understand.

Notice how the measurements are in pixels for example `10px` or `100px`. Pixels are quite small, so you can be quite precise about how big or small you make your margins and things.

You don't have to **declare** attributes for every element on your page, just those where you want to specify (or declare) your choice or preference.

So, your style sheet can be as long (or as short) as it needs to be.

Remember:

When you're writing your style sheets, be very careful with those { and } brackets; also be very careful about the **:** (colon) and **;** (semi-colon) characters.

Get just one of them wrong, and some (or possibly all) of your style sheet won't work.

Just like you save you HTML files with a `.htm` suffix, so your cascading style sheet files need to be saved with a `.css` suffix.

What do I have to do differently in my HTML?

Once you have written your style sheet file, you need to make a few small changes to your Web page. The most important one is tell your browser where to find the style sheet:

To do this, you will need to put a new line of code into the <head> part of your webpage code, something like this:

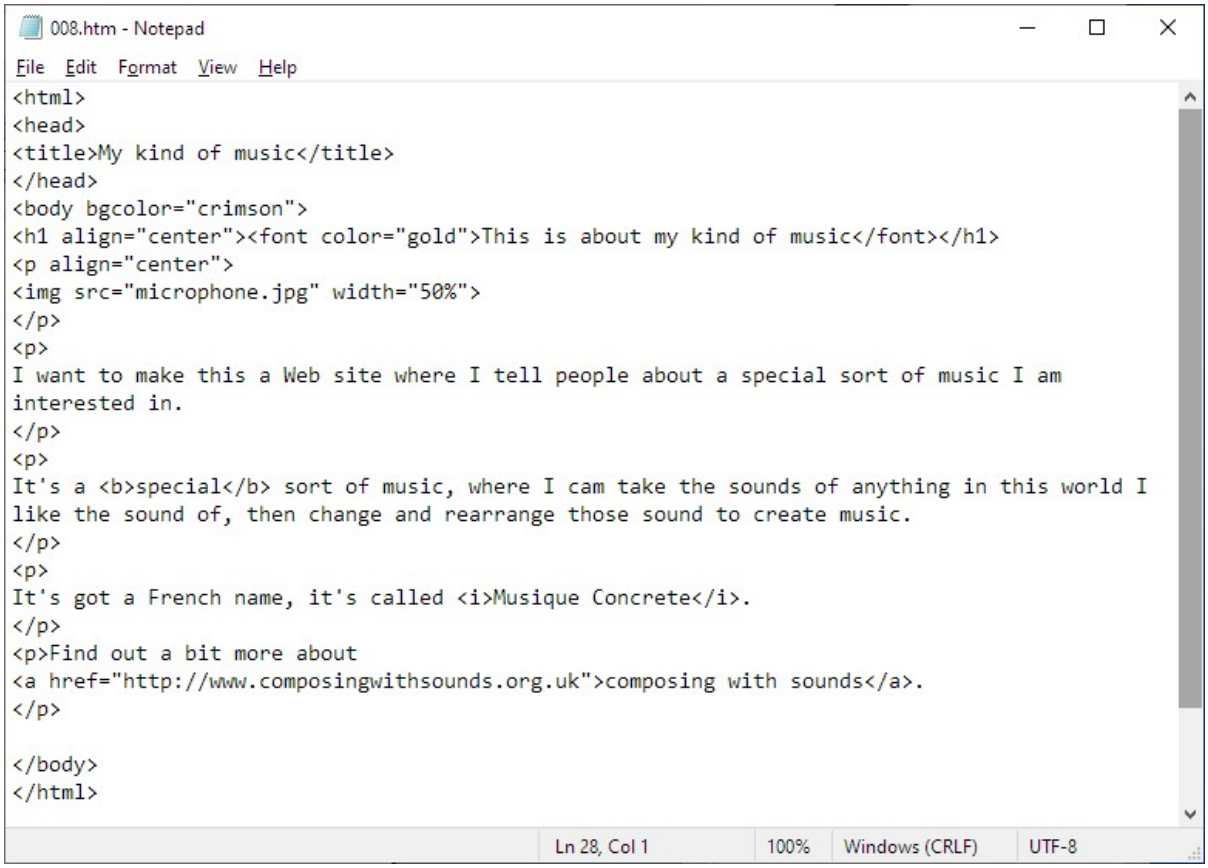
```
<html>
<head>
<title>My kind of music</title>
<link rel="stylesheet" type="text/css" href="mainstyle.css">
</head>
```

`<link>`	tells the browser that we're linking a file to our Web page;
`rel=`	tells the browser what relationship the link file has to the page;
`type=`	tells the browser what type of file it is;
`href=`	just like when we use anchor points to link to other sites, this tells the browser where to find the css file, its URL. I've called mine `mainstyle.css` – you can use any filename, provided it **ends in .css and has no spaces in it**.

OK – That's been a lot to take-in in for one session. Let's have a quick look at an example, before you try this on your site.

CSS Example

Here's the HTML code of a Web page I used as an example a few sessions back (you might remember it):



The most important thing I need to change is to get rid of those `<font>` … `</font>` tags – they went "out" with HTML 4, back at the end of the last century. (They still work because, at the moment, most web browsers are kind enough to support old code.)

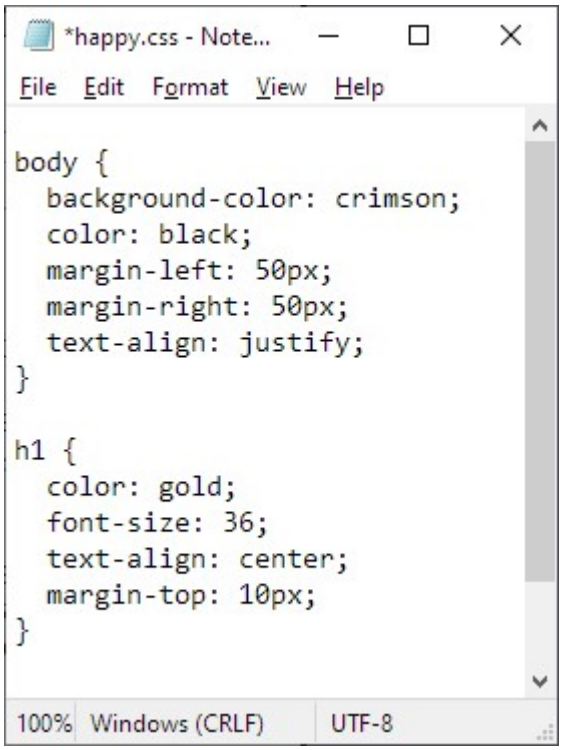
If I'm going to add more pages to my site, it would be good to put things like the background colour, and the properties I have chosen for <h1> headers into that style sheet, too.

This is useful in two ways:

- first it saves me a load of typing in the future;
- second, it keeps all the pages of my site in a nice, neat uniform style.

So, for now I want to declare the attributes for just two elements in my style sheet. Those elements are the main page `body`, and the `h1` header style.

Here's my stylesheet:



I've described how I want the body to look.

`background-color:` describes the colour of the background, while `color:` describes the colour of the text.

I've also decided to give my page some nice, neat side margins, and to justify the text. I hope, when I get to put some more text in my page, this will make it look a little bit more professional.

… and here I've declared the attributes I want the browser to use for everything `h1`.

Now I need to edit my page. It's a good idea, before you start, to save your page again, with a new name – that way, if you wreck it, you still have the old copy, without needing to retype everything. It also means you can compare old and new versions of your page.

Here's the part of my page that I've changed:

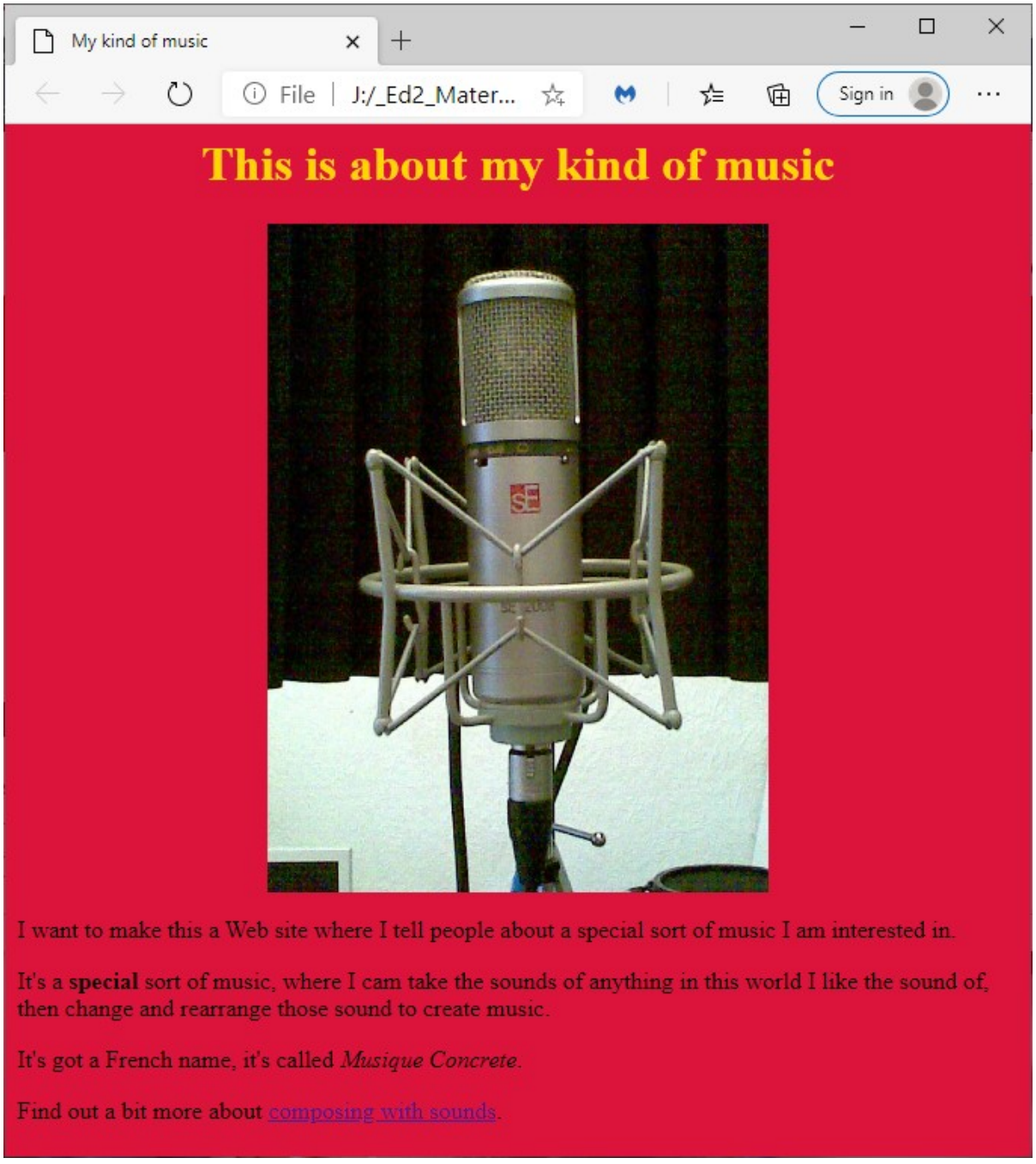
Notice the changes:

- the `<link>` tag in the `<head>` section;
- the `<body>` tag no longer needs the background colour to be names;
- the tags on the `<h1>` header are so much simpler!

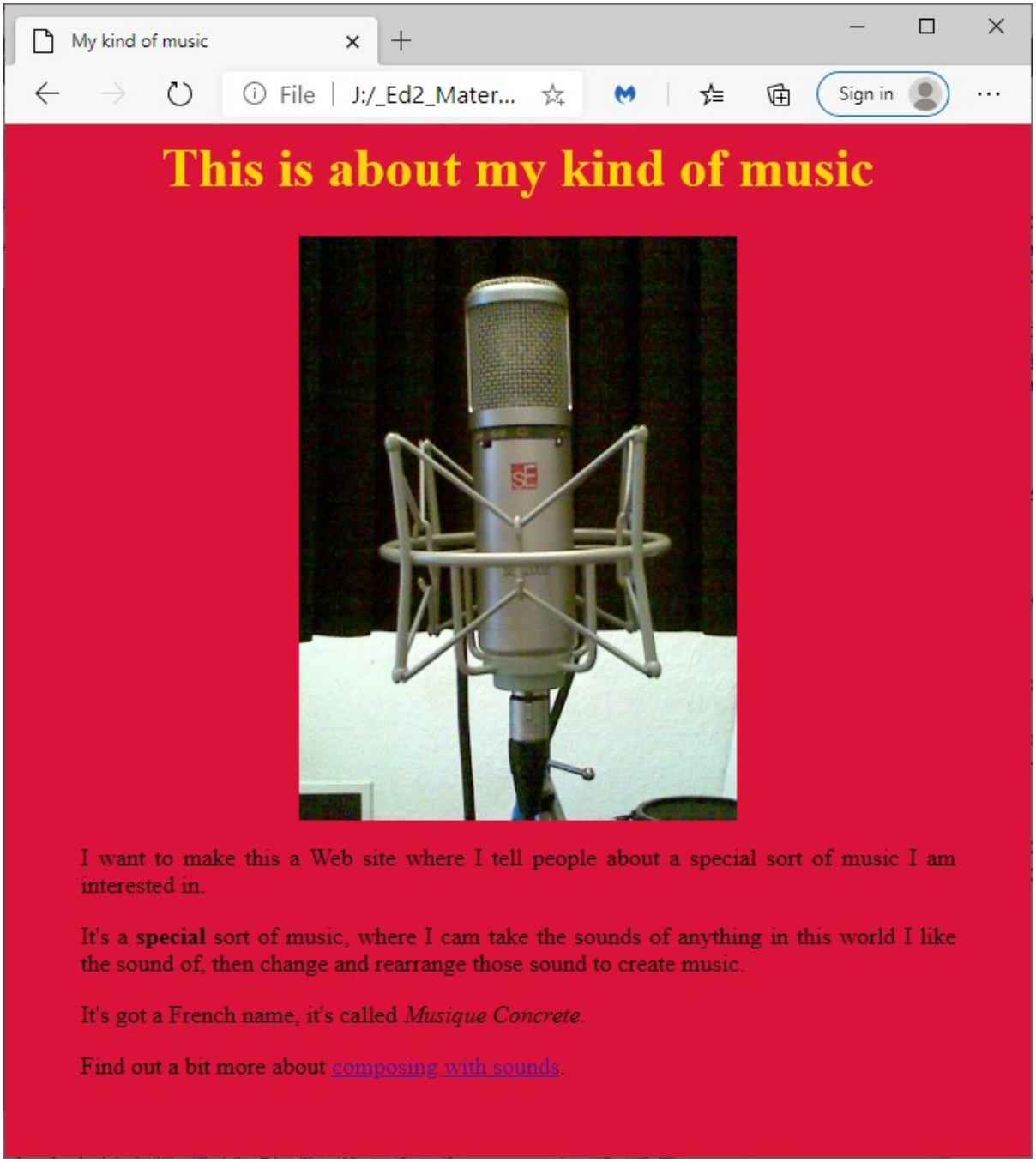
… And what's it done to the way my page looks?

(This is why it's useful to save new versions of your page with different names! I can now look at my old and new pages, to compare them)

Here's the old version:



… and here's the new one:



I don't know about you, but I think those margins make it look a lot more professional.

OK. That's been a lot to take-in in one session.

Have a try, and see what you can do with a stylesheet for your page. Remember to check what I said in the **Remember** box. Be careful to type your `.css` filename correctly when you put the `<link>` statement into your page. Experiment with some of the other elements and attributes. Have fun!

Stay safe until next time, when we'll look at putting additional pages onto your site.