

Web Fundamentals – Module 1

This module takes an initial look at the way that the Web works and the various building blocks that make up a Web Page.

We will start of by covering some core concepts and technologies now, and then later in the course we will dive into some more advanced topics.

Don't forget, if anything is unclear – just ask :-)

Confidential

Overview of the Web

The **World Wide Web**, commonly referred to as '**The Web**' is a worldwide network of computers.

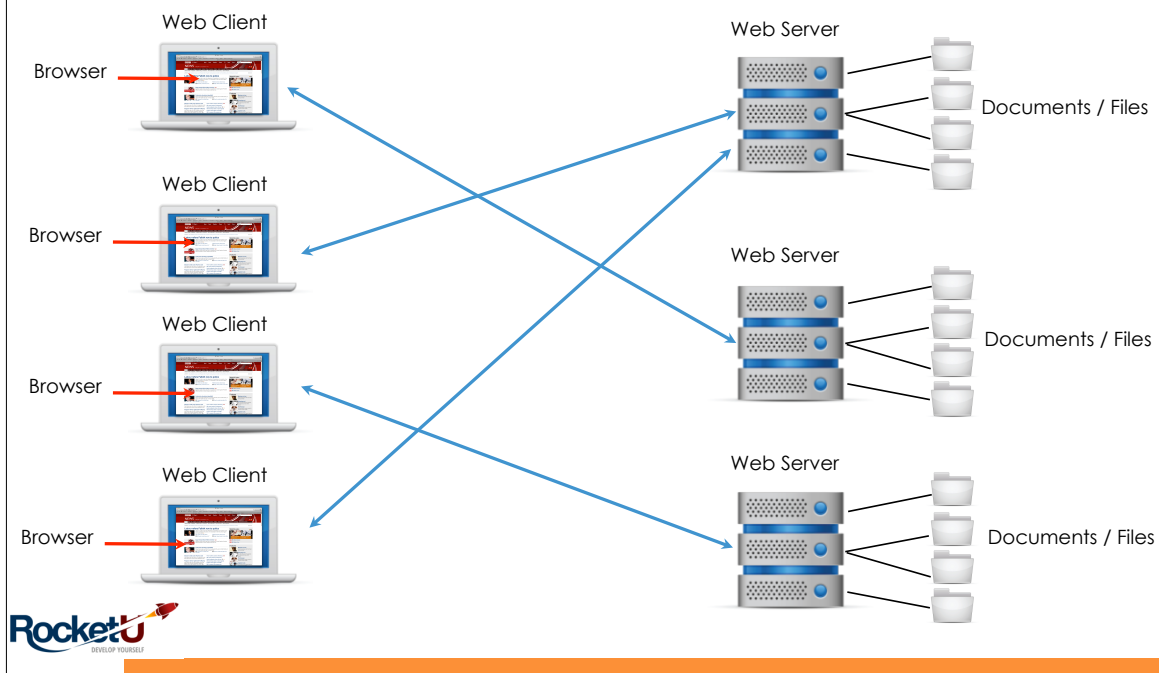
Computers on the Web can share information with each other using a standard communication method known as **HTTP**. We will look at HTTP later in the course.

Documents and **files** that make up web pages are stored on computers that are known as **Web servers**.

Computers that connect to the Web to view sites and pages are known as **Web clients**.

Web clients use **Web browsers** such as Firefox or Chrome to view sites and pages.

Overview of the Web



Web Building Blocks

The Web uses 3 core technologies, or building blocks, to create and display web pages.

- HTML (Hyper Text Markup Language)
- CSS (Cascading Style Sheets)
- JS (JavaScript)

There are other web technologies that are used to varying degrees but all web pages use HTML and the vast majority of pages also use CSS and JavaScript.

Web Building Blocks – HTML

HTML stands for **Hyper Text Markup Language**.

It is a language that was specially created for describing the content and structure of web pages.

HTML is a **markup language**. A markup language uses **tags** to describe content.

HTML is stored in text files and typically each file represents an individual page.

These files are known as **HTML documents**. Each HTML document contains **tags** and **data** (also known as **content**).

We will look at HTML documents in more detail shortly.



Web Building Blocks – HTML

Example HTML Document:

```
<html>
  <head>
    <title>My Web Site</title>
  </head>
  <body>
    <h1>Welcome to my site</h1>
    <p>Hi there, thanks for stopping by...</p>
  </body>
</html>
```

Diagram illustrating the structure of the HTML document:

- Tags** (indicated by arrows pointing to the opening and closing tags):
 - `<html>`
 - `<head>`
 - `<title>`
 - `</head>`
 - `<body>`
 - `<h1>`
 - `<p>`
 - `</body>`
 - `</html>`
- Data (content)** (indicated by arrows pointing to the text between the tags):
 - My Web Site
 - Welcome to my site
 - Hi there, thanks for stopping by...



Web Building Blocks – CSS

CSS stands for **Cascading Style Sheets**.

A **Style Sheet** is a collection of **Styles** that define how HTML elements should be displayed.

Style Sheets can either be incorporated directly into an HTML document or stored in a separate **CSS file**.

Style Sheets that are incorporated into an HTML document are known as **Internal Style Sheets**.

Style Sheets that are stored in separate CSS files are known as **External Style Sheets**.

Both types of Style Sheet can be used by a single HTML page.



Web Building Blocks – CSS

Example CSS Style Sheet:

```
<style>
  h1 { color: blue; }
  p { font-size: 12px; }
</style>
```



Web Building Blocks – JavaScript

JavaScript is a special type of programming language that is designed to be used with Web pages.

It allows pages to be manipulated after they have been displayed by the browser, with effects such as hiding or showing content.

Like CSS, JavaScript can either be incorporated directly into a HTML document or stored in a separate file.

JavaScript code that is incorporated into a HTML document is known as an **Internal Script**.

JavaScript code that is stored in a separate file is known as an **External Script** or **JavaScript Library**.



Web Building Blocks – JavaScript

Example JavaScript:

```
<script>  
    window.alert('Hello World!');  
</script>
```



Web Infrastructure – Browsers

A **Browser** is a special application that is used to display Web pages.

A browser sends a **request** for a web page to a **Web Server** and then waits for the HTML, CSS and JavaScript documents and files to be sent back as a **response**.

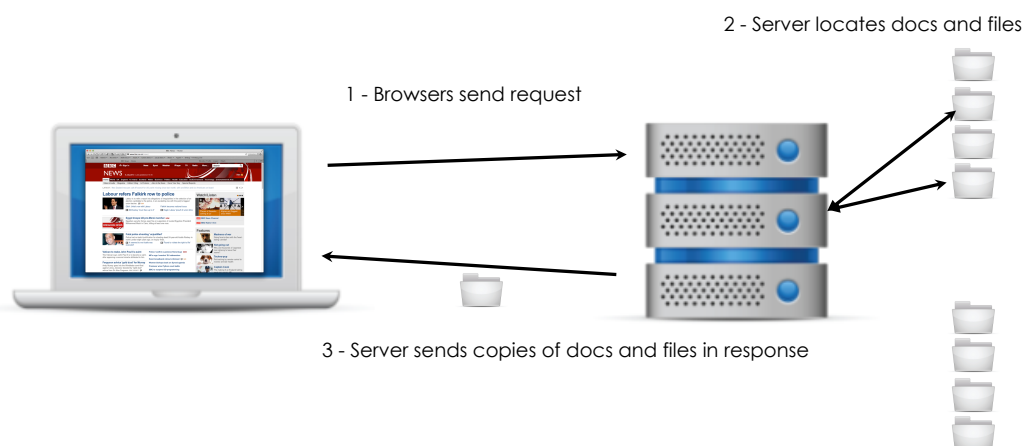
Once the browser receives the various documents and files that make up a page it reads through the HTML markup, CSS styles and JavaScript code and uses them to generate a web page.

Commonly used browsers include:

- ▣ Firefox
- ▣ Google Chrome
- ▣ Internet Explorer
- ▣ Safari



Web Infrastructure – Browsers



Web Infrastructure – Browsers

In order to generate a single web page it may be necessary for the browser to send multiple requests to the server.

The first request will usually result in the HTML document being sent back to the browser.

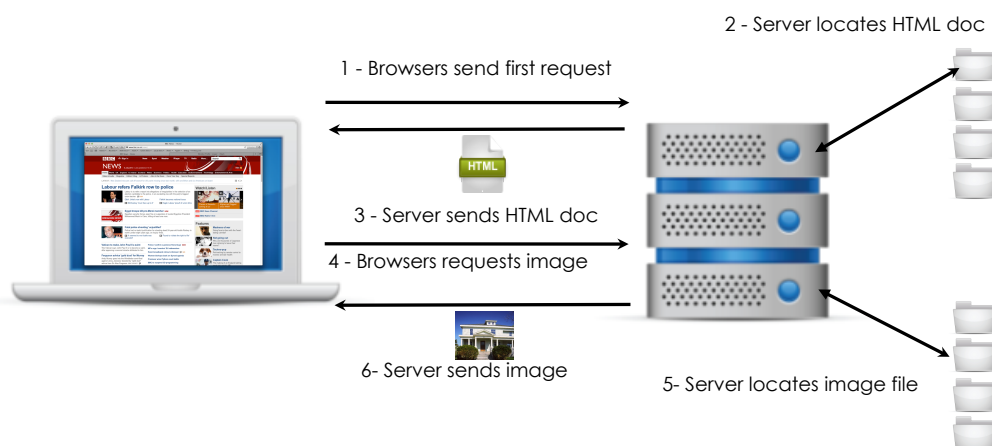
The browser will then start to read through the document and build the web page.

If the HTML document contains a reference to an external file, such as an image, then it will send a further request for that file.

Depending on the type of additional file being requested, the browser may wait for it to be returned before proceeding with the rest of the page.



Web Infrastructure – Browsers



Web Infrastructure – Web Servers

A **Web Server** is a computer that is used to store the documents and files that make up a Web Site.

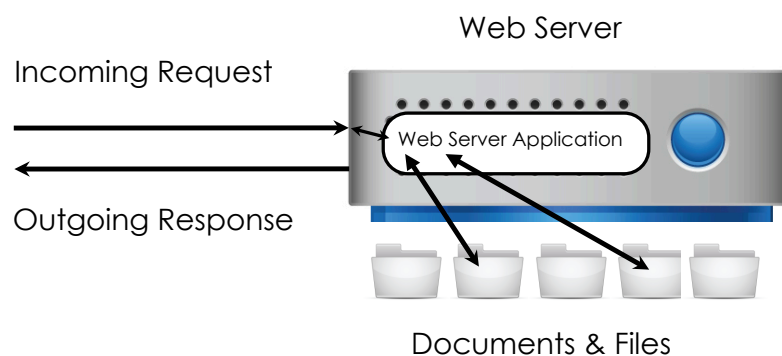
Web servers need to be permanently available via the **Internet** so that Web clients can connect to them to request Web pages.

Web servers use special programs that listen for incoming requests and then locate and send back the appropriate documents and files for a web page.

The most commonly used Web Server program of this type is called Apache. Others include IIS and Nginx



Web Infrastructure – Web Servers



Web Infrastructure – Web Servers

In addition to storing the files and documents for a Web site, a **Web Server** may have other applications installed that are used to store and generate **dynamic content**.

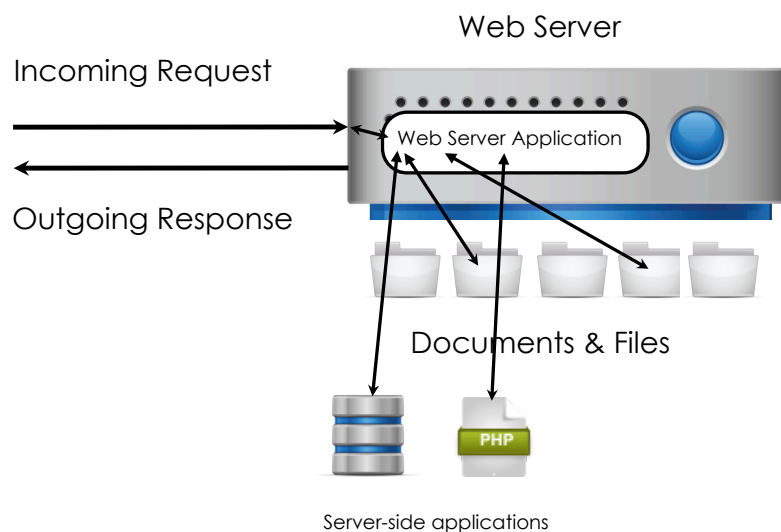
Such applications might include:

- ▣ **Databases** such as MySQL, MongoDB and Oracle
- ▣ **Server-side scripting languages** such as Python, PHP & Ruby
- ▣ **Development frameworks** such as Django, CodeIgniter & Rails

These applications and technologies are covered in the Bootcamp, but we will take a high-level look at some of them in the second week of this course.



Web Infrastructure – Web Servers



Web Infrastructure – HTTP

HTTP stands for **H**yper **T**ext **T**ransfer **P**rotocol.

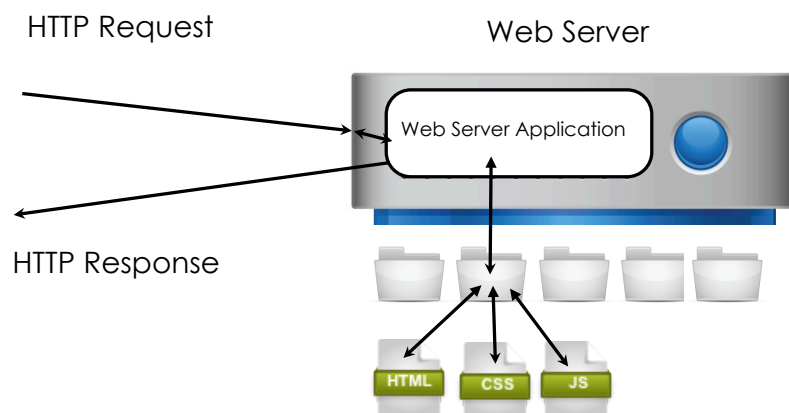
HTTP is a mechanism for sending requests and responses for the documents and files that make up Web pages.

When you click on a link in a Web page an **HTTP request** is sent to the Web Server on which that page is hosted.

The Web Server then locates the relevant documents and files for the page and sends them back to the Web client (browser) in the form of an **HTTP Response**.



Web Infrastructure – HTTP



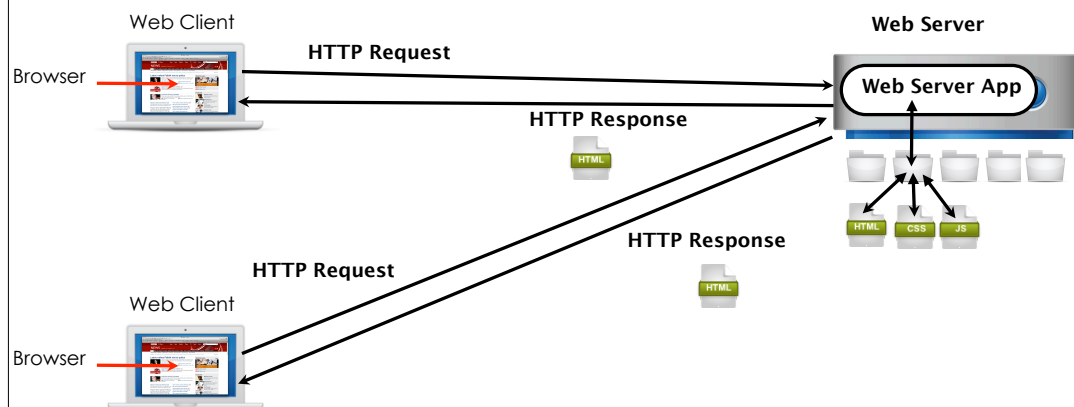
Web Infrastructure – Summary

The processes involved in the display of a web page can be summarized as follows.

- First of all a **Web Browser** sends a **request** for a **Web page** via **HTTP** to a **Web Server**
- When the request is received by the Web Server a Web Server Application, such as Apache, locates the necessary documents and files and sends them back to the Web browser.
- The browser on the web client then processes the HTML, CSS, JavaScript, together with other files such as images, and uses them to display (render) the Web Page



Web Infrastructure – Summary



Web Infrastructure – Local Files

In addition to submitting an HTTP request for a web page via the Internet to a remote Web Server, a browser can open a local HTML document stored on the same computer.

The browser will read an process and HTML, CSS and JavaScript in the document and use it to render a web page.

This can be useful for building and testing simple pages without having to store them on a Web server, but it means that the pages are only available for viewing on the local computer and that certain functionality, such as server-side technologies, are not available.

It is the method that we will use for the first few modules on this course.



Web Infrastructure – Local Web Server

An interim solution that sits between open HTML files locally and sending HTTP requests to a remote Web Server is to configure your computer as a Local Web Server.

This requires applications such as Apache and possibly MySQL, Python or PHP to be installed on your computer and then some additional configuration needs to be carried out.

This can be an extremely useful way of developing websites locally, including the use of server-side technologies, before transferring to a live Web Server for deployment.

We will look at how this is achieved during week two of the course.





The W3C also provides online documentation and education services.

Web Statistics – Browser Usage

2013	IE	Firefox	Chrome	Safari	Opera
May	12.6%	27.7%	52.9%	4.0%	1.6%
Apr	12.7%	27.9%	52.7%	4.0%	1.7%
Mar	13.0%	28.5%	51.7%	4.1%	1.8%
Feb	13.5%	29.6%	50.0%	4.1%	1.8%
Jan	14.3%	30.2%	48.4%	4.2%	1.9%

Source: www.w3schools.com



Web Statistics – Operating Systems

2013	Win8	Win7	Vista	NT	XP	Linux	Mac	Mobile
May	7.9%	56.4%	2.1%	0.4%	15.7%	4.9%	9.7%	2.6%
Apr	7.3%	56.4%	2.2%	0.4%	16.4%	4.8%	9.7%	2.2%
Mar	6.7%	55.9%	2.4%	0.4%	17.6%	4.7%	9.3%	2.3%
Feb	5.7%	55.3%	2.4%	0.4%	19.1%	4.8%	9.6%	2.2%
Jan	4.8%	55.3%	2.6%	0.5%	19.9%	4.8%	9.3%	2.2%

Source: www.w3schools.com



Web Statistics – Mobile Devices

2013	Total	iOS	Android	Others
May	2.64%	1.12%	1.04%	0.48%
Apr	2.24%	1.06%	0.97%	0.21%
Mar	2.46%	1.11%	0.96%	0.19%
Feb	2.17%	1.06%	0.91%	0.20%
Jan	2.18%	1.07%	0.90%	0.21%

Source: www.w3schools.com

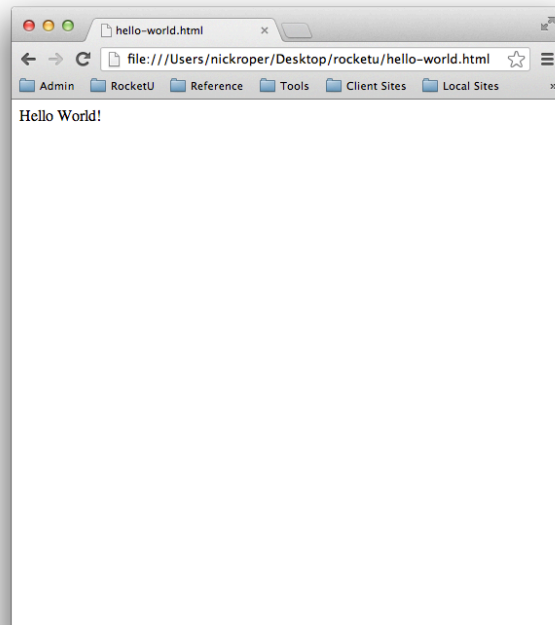
HTML Primer

OK, let's get our first web page up and running.

- ❑ First of all, create a new folder on your desktop named 'rocketu'.
- ❑ Then open up whichever text editor you prefer to use and start a new text file.
- ❑ Enter the text 'Hello World' and save the file as 'hello.html'.
- ❑ Once you have saved the file, start up a web browser and use the File->Open menu options to open the file.

You should see something similar to the next slide...

HTML Primer



HTML Primer

Now edit the 'hello-world.html' file as follows:

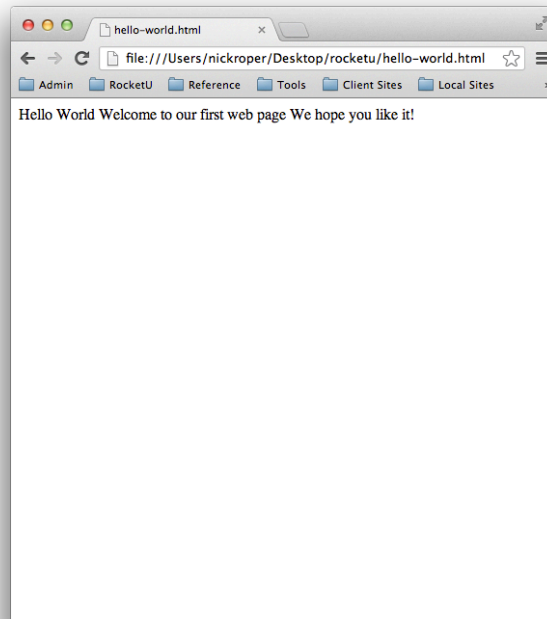
```
Hello World
Welcome to our first web page
We hope you like it!
```

Save the file and then click the 'reload' icon in the browser.

Let's see what it looks like now...



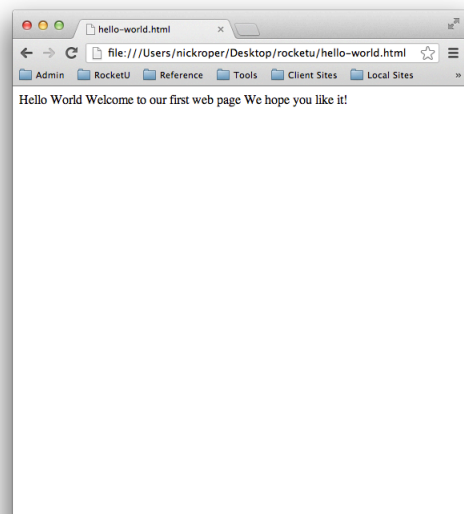
HTML Primer



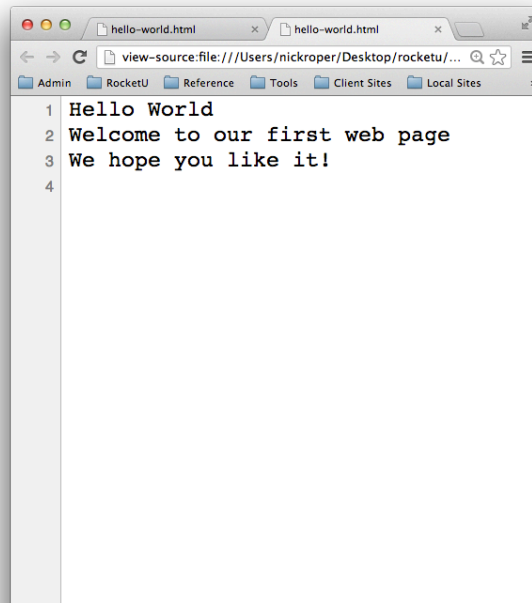
HTML Primer

Hmm, that may not be quite what we were expecting – how come the text is all on one line instead of the three that we entered?

Let's check the text that the browser saw. Right-click on the browser window and select '**View Page Source**'



HTML Primer



```
1 Hello World
2 Welcome to our first web page
3 We hope you like it!
4
```

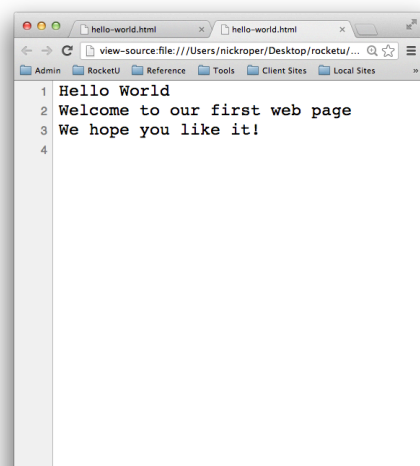


HTML Primer

The 'View Page Source' option lets us see the HTML document that the browser has processed (rendered)

It can be very useful for checking that the browser has seen what we think it should have.

Sure enough, there are three separate lines of text – so why do they appear on one line in the browser?



```
1 Hello World
2 Welcome to our first web page
3 We hope you like it!
4
```

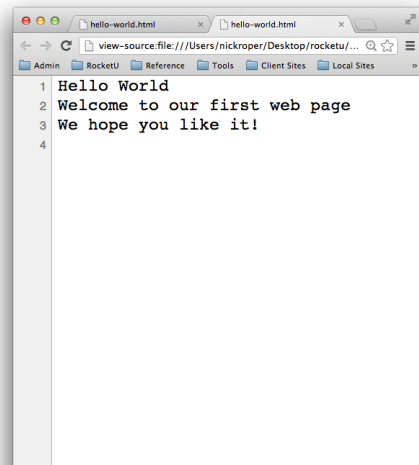


HTML Primer

If you remember, we said that HTML is a **markup language** and that a markup language uses **tags** to describe content.

The issue with our file is that it contains the text/content that we want displayed but there are no markup tags to tell the browser what the text represents.

In other words there is no context to the content, so let's fix that.



HTML Primer

Edit the 'hello-world.html' file once more as follows:

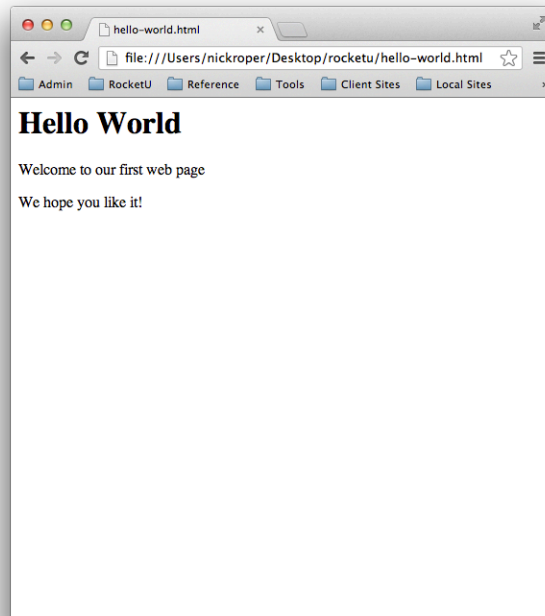
```
<h1>Hello World</h1>
<p>Welcome to our first web page.</p>
<p>We hope you like it!</p>
```

Save the file and then click the 'reload' icon in the browser.

Let's see what it looks like now...



HTML Primer



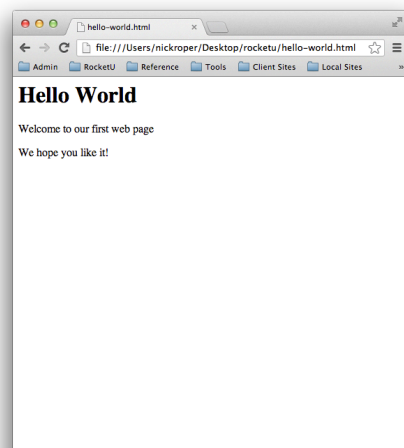
HTML Primer

Now our document has three lines.

By surrounding content with **markup tags** we let the browser know how to treat it.

The `<h1>` **opening tag** tells the browser to treat the content that follows as a **level 1 heading** and the `</h1>` **closing tag** indicates the end of the heading. Similarly, the `<p>` and `</p>` tags indicate the start and end of paragraphs.

Tags plus content form **HTML elements**.



HTML Primer – Summary

In this HTML primer we created our first **HTML document** and opened it directly in a **browser**.

By adding **HTML tags** we were able to provide the browser with **context** as to what the content represents.

An **opening tag**, followed by some **content** and then a **closing tag** makes up an **HTML element**. HTML elements are the components of a web page.

HTML is used to define the pages content and meaning. This is often referred to as **semantics**.

HTML markup should not be used for styling, this is the job of **CSS**.

Note that our HTML document is not properly complete yet – we'll do that in the next module.



CSS Primer

As we said in the HTML primer, the role of HTML is to define the page's content and apply context to it by using tags to indicate where different types of content such as headings and paragraphs begin and end.

Browsers may apply some default styling to different types of content, such as making headings larger than other content, but it is not the job of HTML to control the styling and formatting of the pages content.

The way that content is styled is controlled by the use of CSS Style Sheets. We will cover CSS on more depth later in the course but lets take a quick look at how we can apply some styling to our page.



CSS Primer

Edit the 'hello-world.html' file as follows:

```
<style>
  h1 {font-family: Verdana; color: red;}
  p {color: blue;}
</style>
```

```
<h1>Hello World</h1>
```

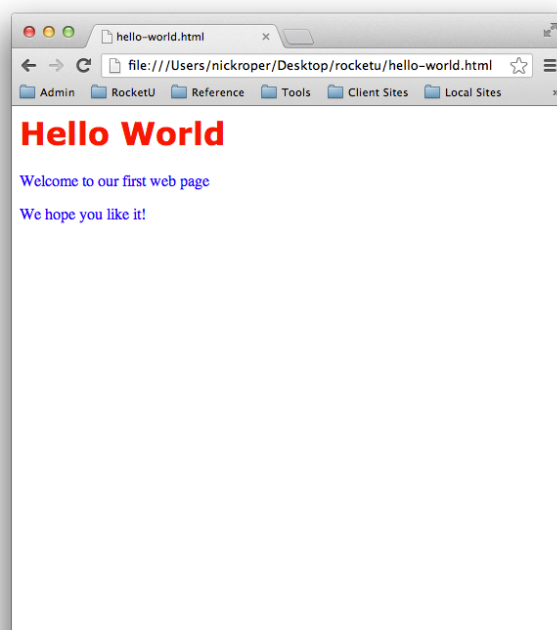
```
<p>Welcome to our first web page</p>
```

```
<p>We hope you like it!</p>
```

Save the file and then click the 'reload' icon in the browser.



CSS Primer



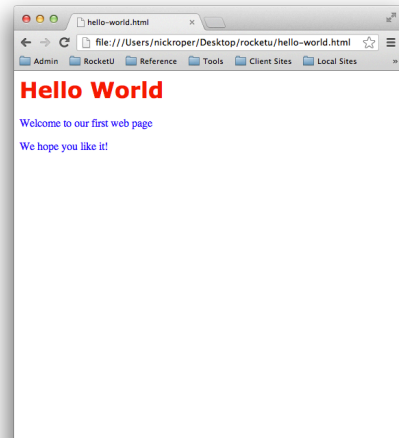
CSS Primer

Now our document has some styling.

The `<style>` and `</style>` tags indicate to the browser that everything in between is to be treated as CSS style rules.

Style rules use a special syntax to identify specific parts of the document and to apply a style to them.

In this case we are telling the browser to make **h1** text red and to use the Verdana font and that **paragraphs** text should be colored blue.



CSS Primer – Summary

The CSS primer introduced the concept of **CSS styles**.

CSS styles are used to apply **formatting to content**.

Styles can be added to a document by inserting opening and closing **style tags** between which one or more **style rules** are specified.

Styles can be added to documents in other ways which we will cover later in the course.

Style rules identify one or more **elements** of a page with a **declaration** of the styles to be applied to them.



JavaScript Primer

So far we have taken an initial look at 2 of the three core building blocks for web pages; HTML and CSS

These are used on just about every web page and, as we have seen, they are responsible to defining and styling a page's content.

The third core technology is JavaScript. JavaScript is a programming language that was created specifically for use with web pages. It can be used for a wide range of purposes and typically it provides some kind of interactive functionality such as hiding or showing content in response to a click and animating text and images to create dynamic galleries and scrollers.

Week two of this course will focus more on the use of JavaScript but for now we will add a simple pop up message when our page loads.



JavaScript Primer

Edit the 'hello-world.html' file as follows:

```
<style>
  h1 {font-family: Verdana; color: red;}
  p {color: #0000ff;}
</style>

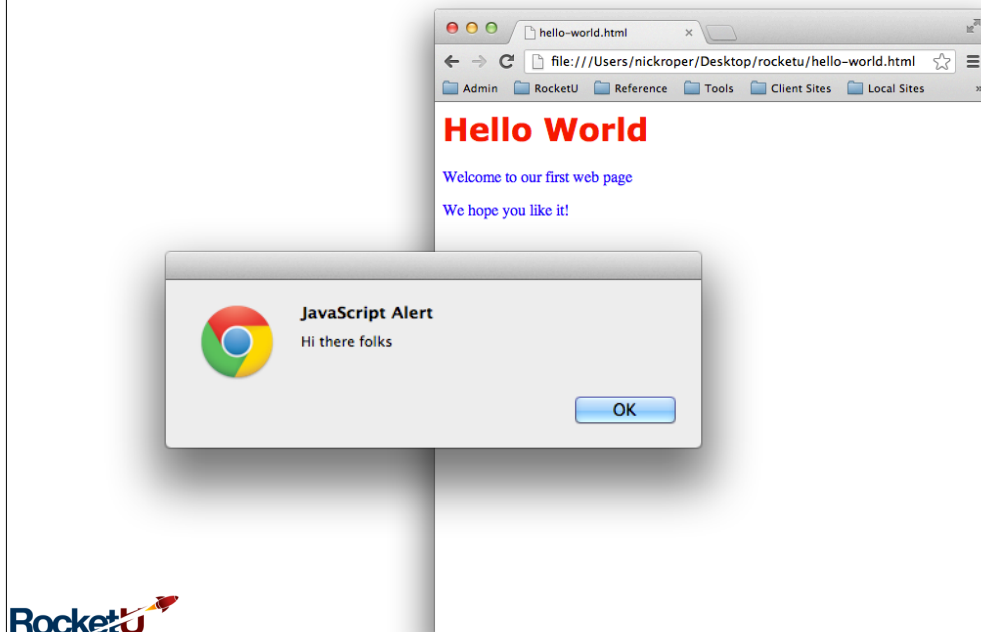
<h1>Hello World</h1>
<p>Welcome to our first web page</p>
<p>We hope you like it!</p>

<script>
  window.alert('Hi there folks');
</script>
```

Save the file and then click the 'reload' icon in the browser.



JavaScript Primer



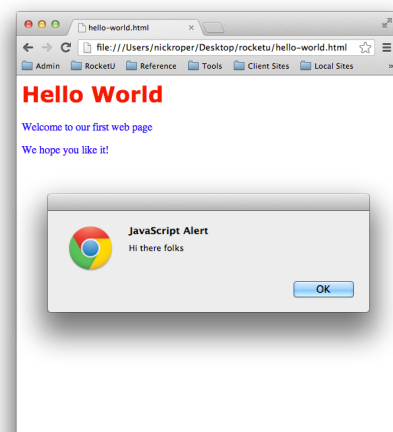
JavaScript Primer

When we reload our page now we see a window appear with a message.

The `<script>` and `</script>` tags indicate to the browser that everything in between is to be treated as JavaScript code.

```
window.alert('Hi there folks');
```

This is a **JavaScript statement** that is executed by the browser and causes an **alert window** to be displayed with a message.



JavaScript Primer – Summary

The JavaScript primer introduced the JavaScript language.

JavaScript is a scripting language that was created specifically for use with web pages..

JavaScript can be added to a document by inserting opening and closing **script tags** between which one or more **JavaScript statements** are included.

JavaScript can be added to documents in other ways which we will cover later in the course.

JavaScript statements can be used to add functionality and animation to a web page.



Module Summary

In this module we took an initial look at the key technologies and processes that are used to generate a web page.

We discussed HTML, CSS and JavaScript; browsers and servers; HTTP and W3C.

Finally we created an initial HTML document and used it to display a web page in a browser. It needs some additional elements in order to be properly complete and we'll do that in the next module.

Next we will move on to look in more depth at exactly how each of these core technologies work. We'll start with HTML and then move on to CSS and JavaScript.



Questions

Do you have any questions before we move on to the next module

?

Don't worry, if you think of something later then just ask!