Data Technologies

Writing Computer Code

Text Editors

You may use any text editor you like, but you will be writing a lot of computer code and there are some features of Crimson Editor that you will find very useful.

A text editor is *not* the same as a word processor. Word processors tend to focus on what a document looks like and try to think for you. The most sophisticated operation in a text editor is cut-and-paste, and even that is usually too dangerous.

Text Editors

Desirable text-editor features for writing code:

- Parenthesis matching
- Automatic indenting
- Syntax highlighting
- Line numbering

Good Coding Practice

HTML will be our first example of creating documents containing computer code.

HTML is the main language used to create documents for the World Wide Web.

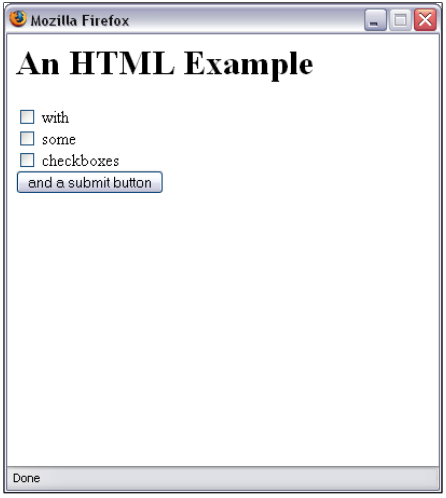
HTML consists of *elements* and *attributes* that describe the layout, appearance, and behaviour of text and graphics in a web page.

A web page can be just a text file containing HTML.

HTML Example

```
<html>
  <body>
  <h1>An HTML Example</h1>
    <form>
      <input name="rb" type=checkbox value="0">
        with<br>
      <input name="rb" type=checkbox value="1">
        some<br>
      <input name="rb" type=checkbox value="2">
        checkboxes<br>
      <input type=submit name="submit"
        value="and a submit button">
    </form>
  </body>
</html>
```

HTML example



HTML Syntax

HTML is a non-proprietary language “standard”.

A group called the World Wide Web Consortium (W3C) publishes “recommendations”, e.g., HTML 3.2, HTML 4.01, XHTML.

We will work with HTML 4.01

HTML Elements

An HTML document is made up of elements, which are usually a start tag and an end tag, with some content in between.

```
<tag>
  some context
</tag>
```

Some elements are “empty”, consisting of only a start tag.

```
<img src="blah.jpg">

<br>
```

In more modern versions of HTML, empty elements are written like this:

```
<br />
```

HTML Elements

The HTML tags “mark up” the content to indicate the structure of a document.

```
<h1>This is a heading</h1>
  This is normal text.
```

Applications that understand HTML, such as web browsers, present the content based on the structural information. For example, headings are drawn bigger and bolder than normal text, and on a line by themselves.

HTML Attributes

HTML Attributes provide extra information about an HTML Element.

Attributes are to Elements as Adjectives are to Nouns.

```
<img src="auckland_uny_logo.gif"
     alt="The University of Auckland">
```

Attribute values should be surrounded by double-quotes (e.g., `src="auckland_uny_logo.gif"`).

HTML Rules

Elements can be nested, but should not overlap.

- This is ok ...

```
<tag1>
  <tag2>
    blah blah blah
  </tag2>
</tag1>
```

- This is *not ok* ...

```
<tag1>
  <tag2>
    blah blah blah
</tag1>
  </tag2>
```

HTML Special Characters

Anything between a `<` and a `>` in an HTML document is treated as a tag.

To put an actual `<` in an HTML document, for example to include something like `age < 50`, you must use `<`. For a `>`, use `>`.

Anything that starts with a `&` in an HTML document is treated as an “escape sequence”.

To put an actual `&` in a document you must use `&`.

Escape sequences can also be used to produce some special characters. For example, `&copy;` produces ©.

The Structure of an HTML Document

There should be a **DOCTYPE** declaration which identifies what version of HTML is being used.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
```

There should be an **html** element.

The **html** element should consist of an **head** element and a **body** element.

The **head** element should contain a **title** element.

A Minimal HTML Document

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>A Minimal HTML Document</title>
  </head>
  <body>
    Your content goes here!
  </body>
</html>
```

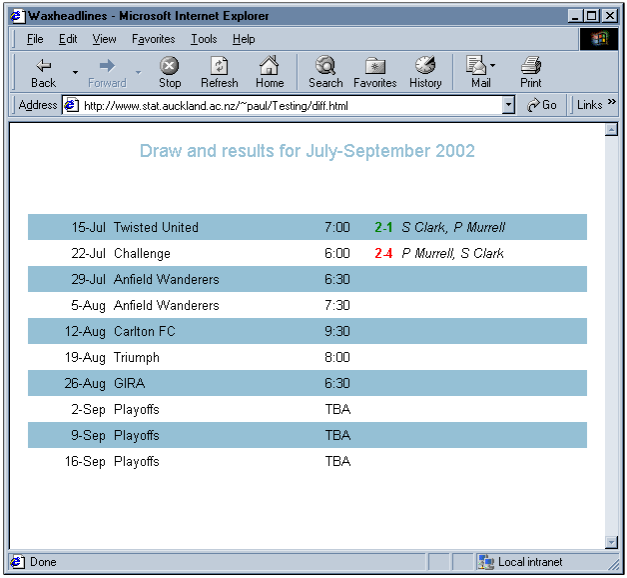
HTML Validation

Most software that implements a computer language acts like a very strict spelling and grammar checker.

Web browsers do NOT act like this.

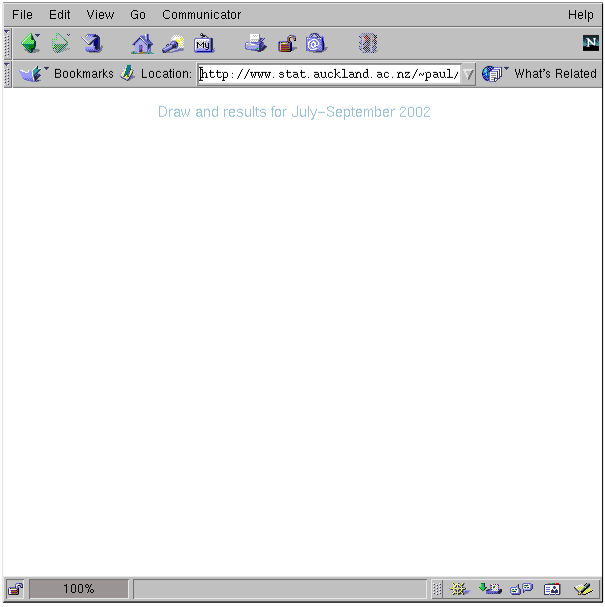
The result is that the same HTML document can look very different in different browsers.

Also, web browsers do not tend to give any feedback if there are errors in the HTML; things just don't look right.



Draw and results for July-September 2002

15-Jul	Twisted United	7:00	2-1	*S Clark, P Murrell*
22-Jul	Challenge	6:00	2-4	*P Murrell, S Clark*
29-Jul	Anfield Wanderers	6:30		
5-Aug	Anfield Wanderers	7:30		
12-Aug	Carlton FC	9:30		
19-Aug	Triumph	8:00		
26-Aug	GIRA	6:30		
2-Sep	Playoffs	TBA		
9-Sep	Playoffs	TBA		
16-Sep	Playoffs	TBA		


File Edit View Go Communicator Help
Bookmarks Location: http://www.stat.auckland.ac.nz/~paul/ What's Related
Draw and results for July–September 2002
100%

HTML Validation

Web browsers are not very good for testing whether your code is correct.

Understanding Errors

HTML Tidy will produce a report telling you about the errors in your document.

This includes important information like what sort of errors there are and where the errors are (e.g., the line number where the error occurs).

A Broken HTML Document

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transi
<html>
  <head>
    A Minimal HTML Document
  </head>
  <body>
    blah blah blah ...
  </body>
</html>
```

File Edit View Go Bookmarks Tools Help

Errors from HTML tidy

```
HTML Tidy for Linux/x86 (vers 1st June 2002; built on May 31 2002, at 23:33:20)
Parsing "/var/www/html/stats220/2006/tmp/phpNIrjpU"
line 4 column 3 - Warning: plain text isn't allowed in <head> elements
line 4 column 3 - Warning: inserting missing 'title' element
line 4 column 3 - Warning: </head> isn't allowed in <body> elements
line 5 column 4 - Warning: <body> isn't allowed in <body> elements

/var/www/html/stats220/2006/tmp/phpNIrjpU: Document content looks like HTML 3.2
4 warnings, 0 errors were found!

To learn more about HTML Tidy see http://tidy.sourceforge.net
Please send bug reports to html-tidy@w3.org
HTML and CSS specifications are available from http://www.w3.org/
Lobby your company to join W3C, see http://www.w3.org/Consortium
```

HTML Tidy modified your code

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 3.2//EN">

<html>
  <head>
    <meta name="generator" content="HTML Tidy for Linux/x86 (vers 1st June 2002), see www.w3.org">

    <title></title>
  </head>

  <body>
    A Minimal HTML Document   blah blah blah ...
  </body>
</html>
```

Good Coding Practice

There are some important principles to learn for writing any sort of computer code:

- Layout your code
- Comment your code

The layout of your code is important for making the structure of the code easy to see. This allows you to navigate the code more easily. It should be easy to read the code.

A good layout will probably look nice; a pretty layout is not necessarily good.

Code Layout

Which of these two pieces of code has the better layout?

```
<html><head><title>A Minimal HTML
Document</title></head><body>
Your content goes here!</body>
```

... or ...

```
<html>
    <head>
        <title>A Minimal HTML Document</title>
    </head>
    <body>
        Your content goes here!
    </body>
```

Code Blocks

A simple rule is to indent “blocks” of code.

“Blocks” mean different things in different languages, but generally have the form:

```
BEGIN
    blah
    blah
END
```

In HTML, open and close tags are good examples.

```
<head>
    <title>A Minimal HTML Document</title>
</head>
```

Long Lines

Sometimes a single line of code can be very long. In such cases, it is better to break the line and indent the continuation. Not ...

```
<table border="1" cellspacing="5" cellpadding="5" summary="just a tabl
```

... but ...

```
<table border="1"
       cellspacing="5"
       cellpadding="5"
       summary="just a table">
```

Code Layout

How much should you indent?

It doesn't really matter. I use 2 or 4 spaces. The only problem with more spaces is that it becomes hard to fit long expressions onto a single line.

Code Layout

Which of these two pieces of code has the better layout?

```
<head>
    <title>A Minimal HTML Document</title>
</head>
```

... or ...

```
<head>
    <title>
        A Minimal HTML Document
    </title>
</head>
```

Code Layout

In addition to indenting code, it is important to use white space (spaces or empty lines) within or between expressions to improve readability.

Using spaces

It is usually better to put spaces around operators, between items in a list, and so on.

```
<table border="1"width="50%"summary="">

<table border="1" width="50%" summary="">
```

Using empty lines

Empty lines can be used to separate conceptually distinct parts of code.

```
<html>
<head>
  <title>
  STATS 220
  </title>
</head>
<body>
  <h1>
  STATS 220<br>
  </h1>
  ...
```

```
<html>

<head>
  <title>
  STATS 220
  </title>
</head>

<body>
   <h1>
   STATS 220<br>
   </h1>
   ...
```

Commenting Code

Virtually all computer languages provide a special syntax for inserting “comments” into code. These parts of the document are not processed.

It is important to include comments so that:

- other people have some idea of how your code works
- *you* have some idea of how your code works.

Commenting Code

In HTML, a comment starts with a `<!--` and ends with a `-->`

```
<html>
  <!-- This HTML document does nothing! -->
  <head>
    <title>
    </title>
  </head>
  <body>
  </body>
</html>
```

Commenting Code

Having no comments in your code is bad. But having too many comments in your code is also bad.

Comments should summarise the code and/or describe the intention of the code. Do not write comments that just repeat the code.

```
<!-- This is the document head -->
<head>
</head>
```

Good Coding Practice

All computer languages have a set of syntax (punctuation) rules which must be obeyed so that your code works correctly.

There is an important distinction between syntax and semantics: syntax means your code follows the rules of the language; semantics means your code does what you want it to do.

This sentence is syntactically correct, but semantically flawed: “Food went to the picnic and ate Paul”.

The computer can tell you whether your syntax is correct, but you must also test whether the semantics are correct.

Good Coding Practice

The techniques for debugging code vary depending on which computer language you are working with.

Here are three important *general* techniques to learn for testing that your code is working correctly:

- Learn to read error information.
- Test your code in small bits and on simple examples where you know what the answer should be.
- Write your own debugging information.

Testing Code

Write and test your code in small pieces.

Build your code up in small steps. Get one step working before adding more code.

```
<html>
  <head>
    <title> Check My Title works</title>
  </head>
  <body>
  </body>
</html>
```

Testing Code

```
<html>
  <head>
    <title> My Title works!</title>
  </head>
  <body>
     Now I can add the body
  </body>
</html>
```

Further Reading

“Code Complete” 2nd Edition, by Steve McConnell
Microsoft Press (2004)