

# I Can Program In HTML!

Thomas Paul Carlson (tpc4@kent.ac.uk)

7th February 2007

## 1 Introduction

The phrase "But I can program in HTML!" is one that is heard worryingly often if you meet the right (wrong?) people. Like all the best ideas, this one came to me while I was drunk. The idea of HTML is to provide a working and usable language that one can program in, while adhering to the original XHTML/HTML 4.0 spec and thus having your programs viewable in a browser. Because the compiler for HTML ignores any tags not in the language it is possible to create pages that look fairly innocuous and contain program code that can be compiled and run. Sort of a crude form of stenography :-).

The language itself contains many features that are in modern programming languages like java and C such as loops, variables (And assignment!), types, conditional statements and functions.

## 2 Language Spec

Please note this is a working copy (Oh the W3C would be proud...) and as such is likely to change. Well. Certain to change.

### 2.1 Starting off...

The opening of an HTML program must be the open HTML tag. Anything else, and the compiler will probably fall over or throw a wobbly.

### 2.2 Magic Value

Immediately following this (The compiler will ignore whitespace) is the title tag. The title for a valid HTML program should be

This page was influenced by rum

This is a magic value and allows the compiler to check that the page you asked it to compile is actually an HTML program rather than just a plain old boring webpage. Note also that the consumption of rum in large quantities is compulsory at this point to gloss over any brain damage that follows in the spec.

### 2.3 Variable Declaration And Assignment

The first thing you are probably going to want to do is declare a variable. For this, you should use the `<acronym>` tag. To declare your variable, do `<acronym title="variableName" id="type">` . Note that this language does not yet have a type system fully decided upon, so for now use either boolean or integer as the type.

You will probably want to do something with that variable, to do that use the acronym tag again as follows: `<acronym title="variableName">NewValueHere</acronym>` . The NewValueHere currently must be either an already declared variable, a number or a boolean (Booleans are defined as true and false).

### 2.4 Comments

You get these for free in HTML, using the original HTMLs comment tag `<!-- This is a comment -->`

## 2.5 Functions

Functions are integral to most programming languages, in this draft they play only a minor part. To keep the HTML file clean and nice, HTML uses `<div>` tags to define a function, an example is `<div id="functionName"><!--Some code should go here --></div>`. Note that this example has no parameters, to add them you should do something like:

```
<div id="functionName" title="(parameter1 , parameter2 , parameter3)">
<!-- Some code here -->
</div>
```

Somewhat amusingly (And confusingly), this use of DIV tags means that one may define functions inside of functions. Exactly how this will work if at all will become clear when the first draft compiler is written and I play around with things a bit more :-).

## 2.6 Anchors, Jumping And Function Calls

Jumping around inside your program is something you will want to do. So, it seems only logical to use the `<a>` tag to do so. To define a place that your program can jump to, define an anchor (`<a name="blah">`). To link to this anchor, just do as you would in regular web-HTML (`<a href="#blah">`). Note that this will only work within the function you are currently in. If you want to jump to a new function, calling it with given parameters you will need to use something similar to: `<a href="#" title="functionToCall" id="(parameter1, parameter2, parameter3)">`. This is a bit messy right now and may well change in the near future...

## 2.7 If statements

Controlling program flow is important! To do this, we use the ordered list `<ol>` element and `<li>` tags to indicate the condition, where to jump on passing the condition and where to jump otherwise. Note that the third (False) condition is not compulsory and the program will just carry on executing whatever follows this if. This allows for nested if statements as well, as the `li` element may contain other lists.

Seeing as the above text is a bit engrishy, heres a couple of examples. Firstly, just a simple if statement:

```
<ol>
  <li title="if (A == 0)">Just some innocuous looking text...</li>
  <li><a href="#jumpIfTrueLocation">Rum is good</a>
  <li><a href="#jumpIfFalseLocation">BLARGH!</a>
</ol>
```

Note that the text can be whatever you want, this way you can hide program code inside nice, normal looking HTML web documents :-). Note that the condition uses the same syntax as the condition in java and you will be able to use all the standard boolean operators.

Okay, so how about something with nested if statements?

```
<ol>
  <li title="if (A == 0)">Whee!</li>
  <li><a href="#jumpIfTrueLocation">heheh</a></li>
  <li>
    <ol>
      <li title="if (B == 1)">Cheese.</li>
      <li><a href="#innerIfJumpLocation">Gorgonzola</a>
      <li><a href="#innerIfJumpFalseLocation">Cheddar</a>
    </ol>
  </li>
</ol>
```

Yuck! Oh well. It works. Note that with all these anchors everywhere, you can probably open your HTML program code in a web browser and take a look at the program flow quite easily.

## 2.8 Echoing To The Console

Well, your program does something. Now how about echoing out to the console to report back to the outside world? Easy. Load up whatever you wish echoing into the magic (global) variable "rum" and then just use the `<br id="magic">` tag. Please note this was hastily tacked on the end. Likely to change.

## 2.9 Terminating Execution

Use the `<hr>` tag to break execution immediately.

## 3 Example Program

Well, now you know the basic programming constructs of HTML. Great! So here is a lump of sample code to grin at.

```
<html>
<title>This page was influenced by rum</title>
<!-- Magic text above :-)-->
<div id="MyFirstFunction">

<acronym title="A" id="integer">Apple</acronym>
<acronym title="B" id="integer">Banana</acronym>
<acronym title="Cheese" id="boolean">Moon?</acronym>

<acronym title="A">1</acronym>
<acronym title="B">2</acronym>
<acronym title="Cheese">A > B</acronym>

<ol>
  <li title="if (Cheese)">The moon is made of cheese</li>
  <li><a href="#yep">Of course it is!</a>
  <li><a href="#nope">What are you on about? :o</a>
</ol>

<a name="yep">
<acronym title="rum">A</acronym>
<br id="magic">
<a href="endMyFirst">Go to the end!</a>

<a name="nope">
<acronym title="rum">B</acronym>
<br id="magic">

<a name="endMyFirst">
<hr>
</div>
</html>
```

So what does that do exactly? Well, we don't yet have an equivalent to a "main" function, so the insertion point is just the first div encountered. A handful of variables are declared and set. The Cheese variable will end up in this code sample being set to false. So, when the if statement chunk of program is run the program will jump down to the nope section and print out the value of B.