

Forum:[Hack Forums](#)

Thread: [Starting with HTML & CSS & How to Start Freelancing](#)

Exemplification - May 30, 2014, 7:45 pm

This thread offers an introduction to HTML, CSS, Freelance Business, and general web development practices and standards.

I do not guarantee that all content in this thread is accurate or is accurate for all readers; please notify me of any errors and I will resolve them.

I apologize for the British spelling of words such as "colour" (color); I had the British dictionary checked at the time and it isn't worth going back through.
All HTML has been validated and tested in Firefox 29.0.1.
I WILL be considering candidate recommendations as standards in this thread.
I HAVE taken some liberties of questionable standards. If you would like to challenge them please private message me

It took 1,366 lines of writing,
an estimated 62,000 characters,
9.4 hours of writing, editing, and condensing,
3 cans of red bull, and 4 soft tacos to write this; it only takes a minute and a minimum of 35 characters to leave feedback; please provide feedback about this thread.

Content SummaryIntroduction to HTMLWhat is HTML?

What are common HTML elements?
How do these elements go together to create a html page?
Create a demo HTML only page.

Introduction to CSSWhat is CSS?
How do these work with HTML to style a page?
Edit our first HTML demo to use CSS for style
Introduction to freelance workWhat is freelancing?
What are the pros and cons of doing freelance work?
How do I start freelancing?
What are some common reasons for freelance failure?
How do I stay safe while freelancing?

Other Content - NOTE: All links are NOT to nsfw[Reading HTML Specification](#)

[Linking to Files](#)

[Getting Your First Clients](#)

[Basic Webmaster Troubleshooting](#) by [Exemplification](#)

[Good resources for starting your website! WebHosting | Domains | HTML | Scripts!](#) by [Nasyr](#)

Content will continuously be added to this thread after the initial posts by me; posts adding content will be linked to in this part of the summary

I need to keep characters available in this post for several links

Character Placeholders [\(Click to View\)](#)[Link Placeholder](#)

[Link Placeholder](#)

[Link Placeholder](#)

[Link Placeholder](#)

[Link Placeholder](#)

[Link Placeholder](#)

[Link Placeholder](#)

[Link Placeholder](#)
[Link Placeholder](#)
[Link Placeholder](#)
[Link Placeholder](#)
[Link Placeholder](#)
[Link Placeholder](#)
[Link Placeholder](#)
[Link Placeholder](#)
[Link Placeholder](#)
[Link Placeholder](#)
[Link Placeholder](#)
[Link Placeholder](#)
[Link Placeholder](#)

Introduction to HTML

What is HTML?

Yes, we have to get over with this part of the lesson but it will be only a few lines. HTML stands for Hypertext Markup Language and is used to structure web pages for humans to read and for bots to view and crawl. HTML by itself is not the prettiest thing; pages made with only html are unstyled and very simple. HTML standards are governed by [W3C \(World Wide Web Consortium\)](#); documentation on all HTML elements can be found on the W3C website. HTML has traveled through several versions and the most current is HTML5 which added several things from previous versions such as offline file storage, addition of form elements, the canvas element, media elements, and a newly developed tight friendship between CSS and HTML.

Here is a view of Hackforums with only HTML ([Click to View](#))

An HTML page is made up of many different elements each with its own purpose; there are more than 100 HTML elements (unless I can't count, which could be a possibility). You'll learn more about elements now.

What are some common HTML elements?

Entering into the world of HTML instantly reveals the structured life it dictates. Each element has its purpose and should be used for that purpose, not as a work around. Just because it works does not mean it is right so before you use an element make sure you are

Here are the common HTML elements used in nearly every site

[b]<!-- Comments! -->[/b]

While not an element, knowing what defines comments is essential to reading this thread. Any text inside of <!-- --> is a comment and will NOT be visible on the page but IS visible if the user were to view the page source. An example follows:

Code [b]<!--[/b] A comment goes in here and will not be visible to the user unless they view the page source -->

<html>

Think of this element as a surname. Your surname is your family and represents your roots, the <html> element represents the root of your HTML document. The use of the <html> element is fairly straightforward, everything other than the DOCTYPE will be housed within this element. An example follows:

Code <!DOCTYPE html>

<html>

[b]<!--[/b] All other HTML content goes here -->

</html>

<head>

Your head element is one of the most important within your html document. It is within <head> that you will direct the browser to find the CSS, JavaScript files, the title of your page (<title>), metadata (<meta>), and some other less used content. This element cannot be valid and contain other text (other than comments) outside of the listed elements. The <head> element comes after the opening of your <html> element. An example of the <head> use follows:

Code <!DOCTYPE html>

```
<html>
<head>
  [b]<!--[/b] Head Content Goes Here -->
</head>
</html>
<title>
```

The <title> element defines the title of your page. This is the name that will be displayed on the top of the browser, tab, or often times on a bookmark. As stated earlier this element goes into the <head> element. An example follows:

Code <!DOCTYPE html>

```
<html>
<head>
  <title></title>
</head>
</html>
<link>
```

The <link> element is used to point toward CSS outside of the HTML document. This element has attributes that must be filled in to link to a stylesheet. You should provide a value for "href" (the location of the file), rel (the relationship to the HTML document, for a style sheet it will be "stylesheet"), type (the type of document, for linking to a CSS file the type will be equal to "text/css"), and media (the media attribute should be set equal to "all" for the purposes of this thread. This will change once the doors to responsive design are opened). Note that the <link> element does not require a closing tag. An example of the <link> element which is pointint to a psuedo document called "styleSheet.css" follows:

Code <!DOCTYPE html>

```
<html>
<head>
  <link href="styleSheet.css" rel="stylesheet" type="text/css" media="all">
</head>
</html>
<body>
```

The <body> element is the main portion of your HTML document. Everything you want to be displayed to the visitor will be within this element. There isn't much to explain with th body, so an example follows.

Code <!DOCTYPE html>

```
<html>
<head>
</head>
<body>
  [b]<!--[/b] Body Content Goes Here -->
</body>
</html>
<p>
```

The <p> (paragraph) element is used to define a paragraph. Content within the <p> element should relate to each other as writing in paragraph is accepted. The use of the <p> element is shown as follows:

Code <!DOCTYPE html>

```

<html>
<head>
</head>
<body>
  <p>The quick brown fox jumps over the lazy dog, but the lazy dog has already feasted on the
eggs.</p>
</body>
</html>

```

<h1>,<h2>,<h3>,<h4>,<h5>,<h6>

The <hX> (0 < X < 7) elements represent headings. <h1> is the most important and defaults to the biggest text in most browsers, and <h6> is the least important and defaults to the smallest text in most browsers. The use of proper headings is important to search engine optimization. Do NOT use heading elements to change the size of font. An example of a heading element follows:

Code <!DOCTYPE html>

```

<html>
<head>
</head>
<body>
  <h1>I'm the most important</h1>
  <h2>I'm a little less important</h2>
  <h3>A bit less important than h2</h3>
  <h4>Even further. At this rate I will be hardly noticable soon</h4>
  <h5>Do I exist? Why am I so unimportant?</h5>
  <h6>I'm the least important</h6>
</body>
</html>
<article>

```

The <article> element is used to define content that is published such as a forum post, blog comment, or a blog post. This element is often found nested within another <article> element indicating that the inner element is in relation to the outer. An example follows:

Code <!DOCTYPE html>

```

<html>
<head>
</head>
<body>
  <article>
    [b]<!--[/b] Perhaps a blog post will go here -->
  </article>
  [b]<!--[/b] Perhaps comments to the post go here -->
</article>
</body>
</html>
<section>

```

The <section> element is a new addition in the world of HTML5 and is a candidate recommendation but not standard. This element is to separate content into logical section. You should not use this element as a container for random content as the <div> element is for; if the <article> element is better suited, it should be used before the <section> element. An example of the <section> element with proper example content follows:

Code <!DOCTYPE html>

```

<html>
<head>
</head>

```

```

<body>
  <section>
    <h1>Snakes</h1>
    <p>Snakes are cold blooded creatures</p>
  </section>
  <section>
    <h1>Rabbits</h1>
    <p>Rabbits are wonderful fuzzy creatures</p>
  </section>
</body>
</html>

```

<header>

The <header> element is fairly self explanatory. This element contains the page header such as the logo and navigational links. This element is NOT the same as the heading elements (<hX>). An example of the <header> element follows:

Code <!DOCTYPE html>

```

<html>
  <head>
  </head>
  <body>
    <header>
      [b]<!--[/b] A logo and navigational content can go in here -->
    </header>
  </body>
</html>
<footer>

```

This is again another self explanatory element. The footer element is at the bottom of the page and contains content such as copyright, extra links, or contact information. An example follows.

Code <!DOCTYPE html>

```

<html>
  <head>
  </head>
  <body>
    <footer>
      <p>&copy; Company 2014
    </footer>
  </body>
</html>
<main>

```

The <main> element is used to house the main part of the page, perhaps articles, weather info, sale info, or contact information could go inside. An example of the <main> element within a page follows.

Code <!DOCTYPE html>

```

<html>
  <head>
  </head>
  <body>
    <main>
      [b]<!--[/b] Most relevant information -->
    </main>
  </body>
</html>
<ol>

```

This is an ordered list and will create a list that uses numbers. You can find an example under .

This is an unordered list that is commonly denoted with bullets. You can find an example under .

This is a list item. This must go within an ordered or unordered list.

Code <!DOCTYPE html>

<html>

<head>

</head>

<body>

[b]<!--[/b] Ordered List -->

List Item 1

List Item 2

[b]<!--[/b] Unordered List -->

List Item

List Item

</body>

</html>

<div>

Used to group content that is not suitable to be grouped with other elements such as section, header, or article. Main purpose is for styling with CSS.

Code <!DOCTYPE html>

<html>

<head>

</head>

<body>

<div>

<footer>

<p>© Company 2014

</footer>

</div>

</body>

</html>

<nav>

The main site navigation. There should be only one per page.

Code <!DOCTYPE html>

<html>

<head>

</head>

<body>

<nav>

[b]<!--[/b] Links ->

</nav>

<footer>

<p>© Company 2014

</footer>

```
</body>
</html>
```

Exemplification - May 30, 2014, 7:46 pm
Let's Create an HTML page!

We're going to create an HTML page and set up the file structure for an organized site. We won't get to styling in this section; styling comes in the CSS section.
Here is what our page will look like once we are complete with the HTML portion. Save your page if you're following along so you can use it in the CSS section.
Our completed HTML page ([Click to View](#))

I would also like to note that the text information in the completed page is not checked for accuracy and does not promote any viewpoints or lifestyle choices. The topic of gods was chosen by the random wikipedia page link. ([Special:Random](#))

As we can see in the image, this page we will be creating is very simple but is made of many of the elements you will find in daily HTML use. It is not meant to introduce all element it is to provide a taste of an HTML page. We will continue to use this sample page in the CSS section, so if you're following along you should save the page for later too :)

Step 1 - Create the Shell

For all HTML documents we will need this shell, this shell will be similar among all HTML pages. Remember from the "elements" section what comments are (`<!--` They look like this if you don't remember `-->`) as you will be seeing a lot of them within these tutorial sections.

The beginning of all HTML documents should declare the "DOCTYPE". Declaring this tells browsers and crawlers what the page is made for. HTML5 has a simple DOCTYPE of "html" The doctype is declared as follows

Code `<!DOCTYPE html>` Simple enough, right?

The rest of the basic HTML shell is fairly self explanatory, and you can find descriptions of all elements in the "elements" section or the following example has the descriptions too. We'll continue on the file that we've added DOCTYPE.

Code `<!DOCTYPE html>`

```
<html lang="en">
```

`[b]<!--[b]` We have to have an HTML element, otherwise there can't be a known HTML page. Also you should set the page language, you can find language ISO codes below this code block. "lang" is an example of an attribute`-->`

```
<head>
```

`[b]<!--[b]` The head element is required and will house items such as `<link>`, `<title>`, and `<meta>``-->`

```
<meta charset="utf-8" />
```

`[b]<!--[b]` Unless you know you need a different charset, keep utf-8 for this guide. `-->`

```
<title>HF Tutorial</title>
```

`[b]<!--[b]` This shows in the browser title or tab `-->`

```
</head>
```

```
<body>
```

`[b]<!--[b]` This element houses everything the user sees (and other stuff, outside the scope of this thread) `-->`

```
</body>
```

</html>

ISO Language Codes(If I forgot one of yours message me and I'll add it) [\(Click to View\)](#)

Language:Code

Abkhazian:AB

Afar:AA

Afrikaans:AF

Albanian:SQ

Amharic:AM

Arabic:AR

Azerbaijani:AZ

Bashkir:BA

Basque:EU

Bengali, Bangla:BN

Bhutani:DZ

Bihari:BH

Bislama:BI

Breton:BR

Bulgarian:BG

Burmese:MY

Byelorussian:BE

Cambodian:KM

Catalan:CA

Chinese:ZH

Corsican:CO

Croatian:HR

Czech:CS

Danish:DA

Dutch:NL

English, American:EN

Esperanto:EO

Estonian:ET

Faeroese:FO

Fiji:FJ

Finnish:FI

French:FR

Frisian:FY

Gaelic (Scots Gaelic):GD

Galician:GL

Georgian:KA

German:DE

Greek:EL

Greenlandic:KL

Guarani:GN

Gujarati:GU

Hausa:HA

Hebrew:IW

Hindi:HI

Hungarian:HU

Icelandic:IS

Indonesian:IN

Interlingua:IA

Interlingue:IE

Inupiak:IK
Irish:GA
Italian:IT
Japanese:JA
JavaneseJ;W
Kannada:KN
Kashmiri:KS
Kazakh:KK
Kinyarwanda:RW
Kirghiz:KY
Kirundi:RN
Korean:KO
Kurdish:KU
Laothian:LO
Latin:LA
Latvian, Lettish:LV
Lingala:LN
Lithuanian:LT
Macedonian:MK
Malagasy:MG
Malay:MS
Malayalam:ML
Maltese:MT
Maori:MI
Marathi:MR
Moldavian:MO
Mongolian:MN
Nepali:NE
Norwegian:NO
Occitan:OC
Oriya:OR
Pashto, Pushto:PS
Persian:FA
Polish:PL
Portuguese:PT
Punjabi:PA
Rhaeto-Romance:RM
Romanian:RO
Russian:RU
Samoan:SM
Sangro:SG
Sanskrit:SA
Serbian:SR
Serbo-Croatian:SH
Sesotho:ST
Setswana:TN
Shona:SN
Sindhi:SD
Singhalese:SI
Siswati:SS
Slovak:SK
Slovenian:SL
Somali:SO

Spanish:ES
Sudaness:SU
Swahili:SW
Swedish:SV
Tagalog:TL
Tajik:TG
Tamil:TA
Tatar:TT
Tegulu:TE
Thai:TH
Tibetan:BO
Tigrinya:TI
Tonga:TO
Tsonga:TS
Turkish:TR
Turkmen:TK
Twi:TW
Ukrainian:UK
Urdu:UR
Vietnamese:VI
Volapuk:VO
Welsh:CY
Wolof:WO
Xhosa:XH
Yiddish:JI
Yoruba:YO
Zulu:ZU

We've done it! We have successfully created the basic shell of an html page. You can save this file as filename.html and you will be able to open a blank document in your web browser except for the title. Here is what it looks like: Blank HTML document ([Click to View](#))

Step 2 - Create an Outline

Creating an outline will make your life easier, planning is a vital part in being successful with anything web related. If you just jump into a project you may discover halfway through that something should have been placed somewhere else; moving these things isn't hard but it is a waste of time and can be very annoying depending on what you are moving. Let's create an outline and then move that outline to a basic HTML template. Here is the outline of what our site will be, feel free to make this template different if you want.

Navigation (Header)
 Modern Gods (<a>)
 Ancient Gods (<a>)
Main Content (<main>)
 Modern Gods (<section>)
 Christian God (<article>)
Hinduism - Brahma (<article>)
 Ancient Gods (<section>)
 Greek Gods (<article>)
 Roman Gods (<article>)
 Comparison (<article>)
Footer (<footer>)
Copyright

So now that we have our basic outline it will be easy to translate that into a HTML template with the elements mentioned in the "elements" section. We have separated the different sections within the main with the section element for the sake of organization, each of the short bits about each type of god in an article element, and then we have used the header and footer elements for thier self explanatory use. The HTML template can be pieced together just like the outline, just add closing tags for each element. Here is what we have created as the HTML template. All of this should go within the <body> element of the page.

Code <header>

```
[b]<!--[b] Logo/Page Name -->
<nav>
[b]<!--[b] Our two links should go here -->
</nav>
```

</header>

<main>

<section>

[b]<!--[b] Modern Gods Section -->

<article>

[b]<!--[b] Christian God -->

</article>

<article>

[b]<!--[b] Hinduism - Braha -->

</article>

</section>

<section>

[b]<!--[b] Ancient Gods Section -->

<article>

[b]<!--[b] Greek Gods -->

</article>

<article>

[b]<!--[b] Roman Gods -->

</article>

<article>

[b]<!--[b] Comparison -->

</article>

</section>

</main>

<footer>

[b]<!--[b] Any Copyright or extra links go here -->

</footer>

Step 3 - Add Content - Header

This step is just as easy as the last. All you have to do is prepare each piece one at a time. Let's start with <header>.

In the header we need two things: the name/topic of the page and the navigation.

The topic of the page is the main idea of the page so the most important heading would suit it best: <h1> (Note, if you place a logo in this section the heading element may not be suited. Use the <h1> element only if it is the focus of the page). The heading goes before <nav> due to both our desired placement and the fact that it is not part of the navigation.

This code is fairly simple:

Code <h1>The Page of Gods</h1>

Moving on to the <nav> we will need to place our two desired links into an unordered list.

Remember an unordered list is `<ul>` and each list item is defined by `<li>`. Go ahead and make a list with two blank items.

Code `<ul>`

`<li></li>`

`<li></li>`

`</ul>`Creating links is easy too! We are going to use the `<a>` element for this and will link to a section on the page. You'll see what I mean later. There is one attribute we must set for `<a>`: `href`. `href` should be equal to a remote destination (E.g. <http://www.google.com/>), or a section on the same page (E.g. `#articleTitle`). Each of our links are going to link to a section of the same page. One will link to the modern gods section (we'll set this section to `#modernGods`), and the other section to ancient gods (we'll set this section to `#ancientGods`). By now you can probably figure out how to make the `<a>` element work, but here is an example anyway.

Code `<a href="#modernGods">Modern Gods</a>`Make the other piece yourself and add it within the two list items in the unordered list.

Code `<ul>`

`<li><a href="#modernGods">Modern Gods</a></li>`

`<li><a href="#ancientGods">Ancient Gods</a></li>`

`</ul>`Just add that list into your `<nav>`.

To make these links go somewhere we must add `ids` (an attribute) to each of our sections like follows:

Code `<section id="modernGods">`Do the same for the Ancient Gods section

This is what the link to a section within looks like when finishedSpoiler ([Click to View](#))

Checkpoint One

We've come quite far, let's make sure we're on the same page.

Here is what we have learned so far. Declaring DOCTYPE

Proper setup of HTML shell

Proper use of `<header>`, `<nav>`, `<main>`, `<section>`, `<article>`, and `<footer>`

Creating links with `<a>`

Creating an unordered list

Here is what our page should consist of.

Code `<!DOCTYPE html>`

`<html lang="en">`

`[b]<!--[b]` We have to have an HTML element, otherwise there can't be a known HTML page. Also you should set the page language, you can find language ISO codes below this code block. `"lang"` is an example of an attribute`-->`

`<head>`

`[b]<!--[b]` The head element is required and will house items such as `<link>`, `<title>`, and `<meta>``-->`

`<meta charset="utf-8" />`

`[b]<!--[b]` Unless you know you need a different charset, keep utf-8 for this guide. `-->`

`<title>HF Tutorial</title>`

`[b]<!--[b]` This shows in the browser title or tab `-->`

`</head>`

`<body>`

`[b]<!--[b]` This element houses everything the user sees (and other stuff, outside the scope of this thread) `-->`

`<header>`

```

<h1>The Page of Gods</h1>
<nav>
[b]<!--[/b] Our two links should go here -->
<ul>
<li><a href="#modernGods">Modern Gods</a></li>
<li><a href="#ancientGods">Ancient Gods</a></li>
</ul>
</nav>
</header>
<main>
<section id="modernGods">
[b]<!--[/b] Modern Gods Section -->
<article>
[b]<!--[/b] Christian God -->
</article>
<article>
[b]<!--[/b] Hinduism - Brahma -->
</article>
</section>

<section id="ancientGods">
[b]<!--[/b] Ancient Gods Section -->
<article>
[b]<!--[/b] Greek Gods -->
</article>
<article>
[b]<!--[/b] Roman Gods -->
</article>
<article>
[b]<!--[/b] Comparison -->
</article>
</section>
</main>

<footer>
[b]<!--[/b] Any Copyright or extra links go here -->
</footer>

```

```

</body>

```

```

</html>This is what the page should look like (Click to View)

```

Take a 5 question quiz: <https://testmoz.com/318206>

Step 3 - Add Content - Articles

Things are starting to become repetitive now that you've learned the basics of structure and how to open and close elements, so this step will go by fairly quickly and you will have the opportunity to do more on your own.

For this part we have five articles and two sections worth of content. The sections are broken up into modern and ancient gods. Sections should group related content and shouldn't become jumbled with anything unrelated.

We've already decided that the sections are within the <main> element, this is due to the fact that our page focuses on gods and some information about the different categories. This hierarchy can be understood with <main> on the bottom (the least specific and defined), the <section> in the middle (getting a bit more specific, in our page we have defined each section to be a different time period), and then on top we have the <article>s.

Each of our sections starts with a heading, the <h2> heading, so lets add that to our first section first.

Code <h2>Modern Gods</h2>After we define the section heading we are ready to jump into the article element and define it's heading as "Christian Gods". We will use the <h3> element for this.

Code <h3>Christian Gods</h3>

Generally in an article the text content comes after the heading, so we're going to place the paragraph of text under the heading. I've generated a small bit of text to add to this article for "Christian gods". The content follows:

Quote:The Christian god is by far the most popular in modern society with many religions worshipping what is believed to be the only god. vague denominations of the Christian religion by popularity are as follows: If you read the elements section and came across the paragraph element you should be familiar with the tags for the paragraph. A paragraph is defined by <p> and should contain text or other in-line elements, or as defined by HTML specifications: phrasing content (See the "other" section at the end of my initial posts to learn more about reading the HTML specifications).

So, after throwing the text into a paragraph it looks like this:

Code <p>The Christian god is by far the most popular in modern society with many religions worshipping what is believed to be the only god. Some vague denominations of the Christian religion by popularity are as follows:</p>Within this paragraph there is some terminology that should have some strong properties: vague denominations. We can add the element because it is "phrasing content" and is allowed in the <p> element. To make the text strong just surround it with the element

Code <p>The Christian god is by far the most popular in modern society with many religions worshipping what is believed to be the only god. Some vague denominations of the Christian religion by popularity are as follows:</p>Add this into your <article> under the heading.

Now we have our paragraph but we need to also add a list ranking the popularity of these religions. We will do this with an ordered list that is defined by (rather than for an unordered list) and has list items determined by as they are with the unordered list. The numbering is done by the browser in a top-bottom format, the top is #1 and continues from there. The finished list will look like this:

Code

Catholicism

Protestantism

Eastern Orthodoxy

Restorationism

Church of the East

Your completed "Christian God" article will look like this.

Code <article>

<h3>

Christian God

</h3>

<p>

The Christian god is by far the most popular in modern society with many religions worshipping what is believed to be the only god. Some vague denominations of the Christian

religion by popularity are as follows:

```
</p>
<ol>
<li>Catholicism</li>
<li>Protestantism</li>
<li>Eastern Orthodoxy</li>
<li>Restorationism</li>
<li>Church of the East</li>
</ol>
</article>
```

Go ahead and create the other articles in both sections; don't worry about the comparisons yet. Here is some sample text to assist you:

Modern Gods - Hinduism - Brahma ([Click to View](#))

While hinduism isn't the most popular religion in the world it is still widely seen in eastern cultures. As understood in Hinduism Brahma was responsible for the creation of the world and all creatures.

Ancient Gods - Greek Gods ([Click to View](#))

Often times ancient Greek gods are used in classes in highschool or college for philosophical purposes. Greek mythology is made up of many different gods.

Ancient Gods - Roman Gods ([Click to View](#))

Like greek mythology, Roman mythology is made of several different gods. These are noted in mythology classes for philosophical purposes.

When you finish it should look something like this:

Spoiler ([Click to View](#))

If it doesn't look like that, play around with it for a while and you will figure out where the mistake was made.

Exemplification - May 30, 2014, 7:47 pm

Step 3 - Add Content - Table

Tables make life better. There is often no better way to organize data than with a table. In previous years tables were used for designing pages, it isn't an option any more (if you do it you will be shunned) but they are still here for data! We're going to use a table to house the equivalent gods of each ancient religion.

Tables are fairly simple and really have four different parts. We have the entire `<table>` element, the table row (`<tr>`) element, the table header (`<th>`) element, and the table cell (`<td>`) element. As can probably be guessed, the table row defines a row. Rows are horizontal (left to right). Each header or cell is a block in that horizontal row. Headers should be placed at the top of the table and data should align properly. We're going to have the heading of Greek and Roman. Under the Greek the Greek name will be in place and under Roman the Roman name will be in place. It will look something like this when finished.

Greek	Roman
Asclepius	Aesculapius
Boreas	Aquilo
Eos	Aurora

So, lets start our table:

Code `<table>`

`</table>`We have created a table! Lets add the four rows we need.

Code <table>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

What about the two headings? Add them to the first row.

Code <tr>

<th>Greek</th> <!--This will be in the top row and the first in the row -->

<th>Roman</th> [b]<!--[/b] This will be the top row and the second in the row.

</tr>

<tr>

</tr>

<tr>

</tr>

<tr>

</tr>

Now that we have the headings we can add the names of the gods as provided above. Keep in mind that the first table cell in the html will be on the left side. I'll add the first two cells, you add the rest.

Code <tr>

<th>Greek</th> <!--This will be in the top row and the first in the row -->

<th>Roman</th> [b]<!--[/b] This will be the top row and the second in the row.

</tr>

<tr>

<td>Asclepius</td>

<td>Aesculapius</td>

</tr>

<tr>

</tr>

<tr>

</tr>

After you finish, save it and check it out. It should look something like this.

Spoiler ([Click to View](#))

Step 3 - Add Content - Footer

We're almost done with the HTML page! All we have left to do is add the footer content which is just text and the copyright symbol. The copyright symbol is the only reason I included a footer in this thread. The copyright symbol should be written with what is called "entities". In other words, you should use © instead of the actual symbol in your writing of a document. You can find a list of other entities here: <http://dev.w3.org/html5/html-author/charref>

Adding the entity is easy, just add the following to the footer and you'll be done!

Code God Info ©

Once finished, your page will look something like this:

Spoiler ([Click to View](#))

Summary

You've learned quite a bit. You've been introduced to the structure of an HTML document, the basic uses of the most basic elements, the proper use of often misused elements.

Perhaps take this short 5 question quiz before you move on: <https://testmoz.com/318268>

Onto the wonderful world of CSS!

Lets add some CSS to that HTML!

I would first like to note that this is NOT a design thread. This web page is not intended to be an influence to an actual design (I am not an artist, and this web page easily reflects that), it is intended to introduce the user to the world of CSS and the many different parts of it. I have purposely used inconsistent habits in this section to give an example of the different ways things can be done. Use whichever you find works best for you, do not have inconsistent files. The CSS in this section is tested on Firefox 29.0.1. The pseudo class selector :not() does NOT work in internet explorer before version 9.

Let me present you with what the page will look like after we add some CSS to it. I've continued with the HTML file from the previous section.

Large Image ([Click to View](#))

As said earlier, don't mind the horrible design. After you read this thread you can fool around with the CSS and make it look better :)

Let's start!

I'm going to break this process into three steps. The first setting up a basic outline, the second setting up the HTML file, and the third actually writing the CSS for our page.

Step 1 - Make an Outline

To make life easier it is recommended to make an outline of what you want the page to look like once finished. This not only improves productivity it is a great chance to get recommendations on the basic layout. Here is an outline I have created of the end page

Wireframe ([Click to View](#))

You should make your own outline, you can use a program like Paint or any of the countless online wireframing tools.

As you can see there is going to be a basic two column layout with headings, a footer, and the two links (The links won't do anything unless the section is out of view).

Step 2 - Prepare the HTML file

This step is also easy considering we correctly used the elements in the beginning section, only a few things need to be added.

First off we need to link to a CSS file that we will be using for this project. You should make the file in the same directory or in a directory withing the directory your html page is located. There are ways around this, but it is not recommended. I've just left the css file in the same directory as the

HTML page and named the file style.css. You can name it whatever you wish.

Linking to a css file is easy, in the previous section you learned about the <link> element and that is what we're going to use. If you remember right the <link> element is located within the <head> element, so that's where we're going to add it.

The <link> element requires three attributes for our linking to a CSS file:

href - location of the css file

type - type of file

rel - the relationship to the HTML file

The complete <link> element is as follows (if you chose a different filename, remember to change it):

Code <link href="style.css" type="text/css" rel="stylesheet">

Now that we've linked the css file we are actually close to being done with the HTML page. There are only two other things we need.

You could see in the outline that we have our content centred on the page, so we need to place all of our content into a container where it can be centred. We are going to use the <div> element for this and will have to assign the attribute "id" so we can modify it with the CSS. Div elements are used to group items together that may not be related in such a way that one of the other section elements should be used.

Code <div id="container">

[b]<!--[b] ALL body content -->

</div>

One more addition!

Footers are sometimes a bit tricky to work with in files that don't have content all the way through the page, so this addition will be used to push the footer to the bottom of the page. I'll explain it a bit more when we get to the CSS part of it. We're going to use another div for this with the class of "push" right after the closing of <main>. This <div> is empty.

Code <div class="push"></div>

Just to ensure we are on the same page, here is the HTML of the entire page:

Code <!DOCTYPE html>

<html lang="en">

[b]<!--[b] We have to have an HTML element, otherwise there can't be a known HTML page. Also you should set the page language, you can find language ISO codes below this code block. "lang" is an example of an attribute-->

<head>

[b]<!--[b] The head element is required and will house items such as <link>, <title>, and <meta> -->

<meta charset="utf-8" />

[b]<!--[b] Unless you know you need a different charset, keep utf-8 for this guide. -->

<link href="style.css" type="text/css" rel="stylesheet">

<!--Link to CSS file -->

<title>HF Tutorial</title>

[b]<!--[b] This shows in the browser title or tab -->

</head>

<body>

<div id="container">

[b]<!--[b] This element houses everything the user sees (and other stuff, outside the scope of

this thread) -->

```
<header>
  <h1>The Page of Gods</h1>
  <nav>
    [b]<!--[/b] Our two links should go here -->
    <ul>
      <li><a href="#modernGods">Modern Gods</a>
    </li>
      <li><a href="#ancientGods">Ancient Gods</a>
    </li>
    </ul>
  </nav>
```

```
</header>
```

```
<main>
```

```
  <section id="modernGods">
```

```
    <h2>Modern Gods</h2>
```

```
    <article>
```

```
      <h3>
```

```
        Christian God
```

```
      </h3>
```

```
      <p>
```

The christian god is by far the most popular in modern society with many religions worshiping what is believed to be the only god. Some vague denominations of the christian religion by popularity are as follows:

```
    </p>
```

```
    <ol>
```

```
      <li>Catholicism</li>
```

```
      <li>Protestantism</li>
```

```
      <li>Eastern Orthodoxy</li>
```

```
      <li>Restorationism</li>
```

```
      <li>Church of the East</li>
```

```
    </ol>
```

```
  </article>
```

```
  <article>
```

```
    <h3>
```

```
      Hinduism - Brahma
```

```
    </h3>
```

```
    <p>
```

While hinduism isn't the most popular religion in the world it is still widely seen in eastern cultures. As understood in Hinduism Brahma was responsible for the creation of the world and all creatures.

```
  </p>
```

```
  </article>
```

```
</section>
```

```
<section id="ancientGods">
```

```
  <h2>Ancient Gods</h2>
```

```
  <article>
```

```
    <h3>Greek Gods</h3>
```

```
    <p>
```

Often times ancient Greek gods are used in classes in highschool or college for philosophical purposes. Greek mythology is made up of many different gods.

```
  </p>
```

```
</article>
```

```

<article>
<h3>Roman Gods</h3>
<p>
Like greek mythology, Roman mythology is made of several different gods. These are noted in
mythology classes for philosophical purposes.
</p>
</article>
<article>
<h3>Comparason in Green and Roman Gods</h3>
<table>
<tr>
<th>Greek God</th>
<th>Roman God</th>
</tr>
<tr>
<td>Asclepius</td>
<td>Aesculapius</td>
</tr>
<tr>
<td>Boreas</td>
<td>Aquilo</td>
</tr>
<tr>
<td>Eos</td>
<td>Aurora</td>
</tr>
</table>
</article>
</section>
</main>
<div class="push"></div>
<footer>
[b]<!--[/b] Any Copyright or extra links go here -->
God Info &copy;
</footer>
</div>

```

```
</body>
```

```
</html>
```

We're done with HTML! Now we get to the fun part.

Intermission

Before we jump into the CSS world lets establish some information about CSS. CSS stands for Cascading Style Sheets and is used to design pages in today's web world. CSS specification can be found on W3C's website. CSS has three main ways (there are more) to interact with an HTML document. CSS can apply rules to all elements of a certain type (or everything with *), apply rules to certain classes, or apply rules to a certain element with a specified ID. This flow of precedence can be written as follows (left = least; right = most): type --> class --> id

CSS denotes classes by preceding them with a period (.) and IDs by preceding them with a hashtag (#). The difference between IDs and classes is fairly simple, an ID is unique to one element (elements can only have one ID) and can only be used once on a page. Classes are not unique and

can be used many times throughout a page and an element can have many classes.

Comments in CSS are not the same as HTML. In CSS comments are denoted with a forward slash (/) and an asterisk (*) to start and end with an asterisk (*) and a forward slash (/). An example comment follows:

```
Code /* A comment can go here */
```

```
/* Comments
```

```
Can
```

```
also span
```

```
multiple lines */
```

Rules are set into the following formats:

Rules for a type:

```
Code header{
```

```
    rules: rules;
```

```
}
```

Rules for a class:

```
Code .class{
```

```
    rules:rules;
```

```
}
```

Rules for an ID:

```
Code #id{
```

```
    rules:rules;
```

```
}
```

Grab your seats, intermission is now over

Step 3 - The CSS - Reset

CSS is hard to teach outside of the formal showing of what each piece does due to the amount of experimentation that is required within it, but we'll give it a shot.

If you remember from earlier, our outline has the page centred. Centring a page is fairly simple. First off we want to do a simple reset in the stylesheet. Resetting the stylesheet allows for more consistent designs and removes many of the browser pre-set margins and padding. There are far more thorough reset stylesheets out there, but this simple one will work for us.

If you have ever researched the internet you probably know that the asterisk is often used to denote a wildcard or all accepting character. In CSS it is similar. The asterisk applies style to all elements on the page. This is a type selector that is overridden by class and id selectors.

The reset is as follows:

```
Code *{
```

```
    margin: 0;
```

```
    padding: 0;
```

```
}
```

Congratulations, you have successfully reset. Let's continue

Step 3 - The CSS - Body

Let's just do two quick things to the body. Let's change the background colour and set its height to 100% so it takes up the entire page. I have chosen a random HEX color: #A6CBF5. Changing these two properties is fairly self explanatory.

```
Code body{
```

```
    background-color: #A6CBF5;
```

```
    height: 100%;
```

```
}
```

[b]Step 3 - The CSS - Container[/b]

You added a div with the id of "container" in step two, now lets style it! Most screens today can handle a 1000px wide page, so we're going to set the container to 1000px (use "width" for this), we also want text to be aligned on the left for readability so we need to assign that (text-align), and finally we want to centre the container in the middle of the page, so we'll set the "margin" property for this.

When setting the margin you will often find people writing it in shorthand. The shorthand looks like these "margin: 10px 22px 33px 44px", "margin: 10px 22px, 33px", "margin: 100px, 200px", or "margin: 0" which mean "top, right, bottom, left", "top, right and left, bottom", "top and bottom, right and left", and "top, bottom, right, left", respectively.

These shorthand methods can easily be replaced by using "margin-left", "margin-right", "margin-top", and "margin-bottom". Many beginners will opt for not using the shorthand because of the ease of readability.

Now that we have the three things we need to set lets place them into CSS language. Try and see if you get it right before you look at the code.

[code]

```
#container{  
  width: 1000px;  
  text-align: left;  
  margin: 0 auto;  
}
```

Exemplification - May 30, 2014, 7:49 pm

Step 3 - The CSS - Nav

The <nav> will be the most intense part of CSS for this thread. As you could see in the outline the links are horizontal but still in an unordered list, the navigation is on the right side of the page, and there is a space between the two items. Lets start with placing the <nav> on the right side of the page. We're going to use float this time. This will place the <nav> as far right as it can go within its container (that we have made 100px wide). As you can probably guess we're going to set "float" to "right" for the <nav> element. The reason we didn't set an ID to the nav and are using a type selector is because there is only one <nav> allowed per page.

We can set it as follows:

```
Code nav{  
  float:right;
```

}It floats! If you save and view your HTML file the nav should be on the right side as shown below.

Nav on Right ([Click to View](#))

What about those ugly bullets? Lets get rid of them!

We're going to be using something called a "descendant selector". These selectors point toward elements that are within other elements. In our case we are going to target the unordered list within <nav> as if we were to use another unordered list somewhere we don't want it to be without bullets and inline. A descendant selector can be identified in the CSS file by the listing of types, id, or classes.

Targeting the unordered list in the nav can be shown as follows:

```
Code nav ul{  
}
```

What exactly do we want to do to the list as a whole (not including list items)? Upon thinking about it we only want to do three simple things to the list: have no style (bullets), have no margin, and have no padding. We already know how to eliminate the margin and padding and we have already done it in the reset piece at the beginning of the CSS. So all we have to do is to eliminate the bullets! For this we are going to use "list-style-type" and set it to none.

```
Code nav ul{
```

```
    list-style-type: none;
```

}If you save and view your html file you should see that you have two links that are now without bullets. Now we need them to be side by side instead of bottom to top. For this we are going to befriend the (list item) element and change how its method of displaying to inline. We are going to target the within the that is within the <nav> which is denoted like follows:

```
Code nav ul li{
```

```
}Setting the display to inline is as simple as
```

```
Code nav ul li{
```

```
    display: inline;
```

```
}Your list items are now side by side!
```

They seem a bit too close to be comfortable, let's give them some room to breathe. We're going to place a left margin on the second list item so there is a space between the two. We, however, don't want the first item to have a margin too so we're going to use what is called a pseudo selector. Please understand that this does not work in <IE9>. You will find more information about pseudo selectors in the "Other" section.

To select the list item that is not the first we will be using :not(:first-child). This means that it will select all list items that are not the first. Then we are just going to add a margin on the left side! It looks like this:

```
Code nav ul li:not(:first-child){
```

```
    margin-left:50px;
```

```
}
```

Think we're done? Well, we're not. Lets change the link decorations, change the hover effect, and change how they're displayed. We're going to use more pseudo class selectors (that surprisingly actually work in IE) for this: hover, active, and visited. First lets style the general <a> to have no decoration (underline), a colour of black (black), and display to "inline-block". Keep in mind we're still targeting within the nav ul elements.

```
Code nav ul a{
```

```
    text-decoration: none;
```

```
    color: black;
```

```
    display: inline-block;
```

```
}
```

I think we should make the block have a background colour of black when hovering and the text colour to turn white, so lets use the :hover pseudo class.

```
Code nav ul a:hover{
```

```
    color: white;
```

```
    background-color: black;
```

```
}
```

Now I want to make the text turn black when he have the mouse button held down on the link, so we're going to use :active.

```
Code nav ul a:active{
```

```
    color: black;
```

```
}Now we are finally done with the <nav>.
```

Step 3 - The CSS - Main

The main element is going to be really simple to style. All we're going to do is add a small top margin to it so there is space between it and <nav>.

You can figure this one out; just set the margin-top to some value. I'm going to use 5%.

```
Code main{
```

```
    margin-top: 5%;
```

```
}
```

Step 3 - The CSS - Section

We have divided our main into two sections, one for modern gods and one for ancient gods. We have already assigned appropriate ids to these sections. Since we want a two column design we will have to set it to 50% width and float the modernGods section left. This can be done as follows.

```
Code #modernGods{
  width: 50%;
  float: left;
}
```

Step 3 - The CSS - Articles

For the articles all we want to do is add a bit of space between them, so we'll set the top margin.

```
Code article{
  margin-top: 2%;
}
```

Step 3 - The CSS - Table

The table will be quick. Lets set the table, the table header, and the table cells to the same border of a solid blue 2 pixel line:

```
Code table, th, td{
  border: 2px solid blue;
```

}Note this difference between descendant selectors. Using commas in this situation separates the elements making each of them have the same CSS and not targeting <td> within <th> and <table>.

Step 3 - The CSS - Footer

Footers are tricky assholes, there are some simple solutions to fix this triciness. We're going to be using one that has become more and more popular. We're going to have to decide how large we want our footer to be; 80 px seems nice to me.

We have to take a couple paces back to the #container. We need to add two things to this: min-height and an edit on the margin.

We need to change the margin to have a negative for the bottom margin so we'll use this format: 0 auto -80.

The finished #container should contain this:

```
Code #container{
  width:1000px;
  text-align:left;
  min-height: 100%;
  margin: 0 auto -80px; /*Top, left & right, bottom */
}
```

Now lets actually move to the footer and that mysterious "push" div we created earlier. We need to clear the float on both of the so they aren't negatively effected by a misplaced or misused float. We do this with "clear". Then we need to set the height to the opposite of the #container bottom margin: 80px.

This can be achieved like follows:

```
Code footer, .push{
  height:80px;
  clear: both;
}
```

Now we need to do some editing of the footer itself. If you look at the completed page I have the footer with a background of red and the text vertically centred.

Vertically centred text doesn't have something like "text-align". We use line height for this. Line height makes the text in the middle of, well, lines; think of notebook paper. If we make the line height the same as the height of the footer it should centre the text vertically.

```
Code footer{
  background-color: red;
  text-align: center;
```

```
    line-height: 80px;
}
```

Step 4 - OMG

Sit back, grab a cold beer (Underage kids you better be drinking root beer); you're done. We have completed the CSS part of this quest. The final CSS should look similar to this:

```
Code *{
    margin:0;
    padding:0;
}

body{
    background-color: #A6CBF5;
    height: 100%;
}
#container{
    width:1000px;
    text-align:left;
    min-height: 100%;
    margin: 0 auto -80px; /*Top, right, down, left format */
}

header h1{

}

nav{

    float:right;

}
nav ul{
    list-style-type: none;
    margin: 0;
    padding: 0;
}

nav ul li{
    display: inline;
}

nav ul li:not(:first-child){/*please note that this is NOT valid for any internet explorer under 9 */
    margin-left: 50px;
}

nav ul a{
    text-decoration: none;
    color: black;
    display:inline-block;
}

nav ul a:hover{
    color: white;
    background-color: black;
```

```

}

nav ul a:active{
    color: black;
}

main{
    margin-top: 5%;
}

#modernGods{
    width: 50%;
    float: left;
}

article{
    margin-top: 2%;
}

table, th, td{
    border: 2px solid blue;
}

footer, .push{
    height:80px;
    clear: both;
}

footer{
    background-color: red;
    text-align: center;
    line-height: 80px;
}

```

And the finished product should look like this:
 Final product [\(Click to View\)](#)

We shan't dwell on this section any longer!

Freelancing!
 This section of this thread can be independent or attached, depending on your path. It was decided at the last minute to add this part. Much of this information is my opinion.

I've created this section because I have reached success with freelancing and want to share some of the common pitfalls and share some things to do to ensure a swell time.
 Before I officially get started I would like to explicitly state that the information within this thread is NOT proper legal advice and shouldn't be treated as such. Any legal material in this tread is only a guidance path and you should consult with a legal expert before embarking on any questionable legal adventure.
 Let't get started.

What is freelancing?

Freelancing is providing services as a person without a business. Freelancers can be found in fields such as software engineering, design, networking, and many more. The term "independent contractor" is often used in place of freelancer.

Freelancing isn't just that. I encourage anyone who has the ability to become a freelancer to become one. It isn't something you will want to do forever, but it is a wonderful way to learn about what is important in life and teaches very important skills.

What are the pros and cons of freelancing?

Freelancing, as stated above, is a wonderful thing to be able to do for income. You will learn quite a bit about life and yourself during this time in your career because of the very independent nature of the work. I'm just going to throw the pros and cons into a list.

Pros: You're on your own. You have no boss but yourself, your contract, and perhaps your morals. You can choose if you want to work today or if you want to sit on the couch eating cheetos.

You can have as much or as little responsibility as you want. You can provide services maintaining material for your clients or you can send it to them and never hear from them again. You can take projects that are a challenge or you can take projects that are little work.

Your advancement is unlimited. You can expand your freelancing quest into a successful business with the right mindset.

Cons: You're on your own. It is a pro too, but being on your own can have poor effects. The social side of work is something that you miss out on, you also are responsible for actually working and producing an income. Poor motivation might end with you out on the street with an eviction notice on your door.

You're responsible for all those accountant duties that would be done for you in the public workforce. You have to calculate and pay your taxes, you have to find your own health insurance, you must put money away yourself for retirement.

You're responsible for yourself. Unlike working for a Fortune 1000 company you are responsible for your legal duties. If you fuck up and you don't have the right material signed you could be responsible for damages that greatly exceed your yearly income.

How do I start freelancing?

The primary reason for me adding this section to this thread to to advise people how to start freelancing, and how to be prepared.

Far too often I find people in the local community that are not prepared to do what they are asked. They do not know what they are doing. They, like some of you, have only read a few online tutorials and are trying to sell their services for expert prices. These guys nearly always fail. (The ones who don't fail are generally without morals).

Getting started freelancing is easy. First off you should know what your doing. If you want to start designing and selling web sites as a freelancer don't start until you can do nearly everything you could possibly be asked. If you can't answer 98/100 questions the client has about your service you are not ready to freelance. Know your material. Know what you are attempting to provide. If you can't provide a complete site in 30 days like you said you could, don't start freelancing.

Become familiar with your personal abilities and limits. You know what you can and cannot do. There is no shame in not knowing how to do something, just don't say you can if you can't.

Become social. I know many people who desire to be freelancers are the social awkward types, some may even proceed to be socially anxious. If you are one of these two and are unwilling to change you will fail at freelancing. You must be able to market your services; this marketing sometimes results in you going door to door finding businesses without "no solicitation" signs and speaking to the owner. You may even be placed in situations where you are required to present

yourself to a board of people.

Become a predator. I don't mean going out and finding little children to stick in your van. Learn who your competition is in your community. Learn what they are charging and what they offer. Offer this for less than them and of better quality. You may not be able to sustain yourself doing this but you can take business away from others. Once you have the clients they generally stay with you if you can be more convenient than another. This predation will gain you experience too. It is a bloody world out there, learn to live off of blood.

Become confident. You will not be successful if you are not confident. Tell the client what they need, sell yourself. Then upsell yourself. If the client doesn't need something make them need it.

Stand your ground. Once you gain clients make it hard for them to go to another person. You don't even have to go the route of trapping them, make them want to stay. Offer them discounts. Offer them chances to get their site re-evaluated for free. Go visit them in their business and ask how everything is working for them. Find something else that they don't have that they could use.

What is the most common reason for freelance failure?

The number one reason for failure in freelancing is the lack of knowledge. People read tutorials like the one above and think they are ready to start freelancing. The first project that isn't straightforward throws them for a loop and they cannot finish it. A business owner is not going to go back to someone who didn't know what they were doing to start with. Learn your material and keep learning. This isn't a business that is the same year after year. Everything changes, trends come and go, and you have to change.

How do I stay safe while freelancing?

If you plan on doing freelancing as a career you must have some legal assistance. You need a document stating what you are providing and what the client requests. Not having this will have you in hell.

If you can't afford a lawyer starting out there are countless freelance documents available on the web. With some minor modification they are easily used. Note every item that is to be provided on your document to be signed. Leave nothing out. Make sure you can provide all of these items.

Other

Reading HTML Specification

The easiest way to find the information is on Mozilla developer.

Here we have the page for <nav>. We can see the categories that it is part of and the categories that are allowed inside of it under "permitted content". It also states that there must be a start and end tag. "Permitted parent elements" denotes the requirements for the element <nav> is placed in HTML <nav> Spec ([Click to View](#))

Matrix Programmer - May 30, 2014, 7:58 pm

WOW. I just want to say this in an AMAZING tutorial. Thank you for this a lot! Keep up the good work mate!

IMadeYouReadThis - May 31, 2014, 6:52 am

Amazing, The amount of effort, Time and Thought that must have went in this guide is amazing! You sir are a legend I'm sure I will not be the only one who will benefit from this guide thanks.

~IMadeYouReadThis~ ?

potatopete54 - May 31, 2014, 7:31 am

<http://prntscr.com/3obvkv>

If you have time, you can also explain the attributes for each tag. Nice tutorial, but more-of 'tl;dr' imo (no offence). I noticed some sentences which made no sense, will point it out later onwards, very sleep now. Good job! ?

Nindo - May 31, 2014, 7:34 am

Hmm nice tutorial, i already know the basic's but this can help people

ObscureCoder - May 31, 2014, 7:36 am

Apart from how some of the code could be adjusted this is a VERY good tutorial. I recommend it to anyone starting out.

After someone covers this they should cover PSD to responsive HTML & CSS.

OP - Looks like you put a ton of effort into this, well done.

You're using an outdated part of HTML though; tables. No-one uses them on sites anymore, it's floating divs now.

I know you said they are not being used for design but they shouldn't be used for anything really.

tl;dr - You said you used questionable standards.

Exemplification - May 31, 2014, 8:56 am

(May 31, 2014 07:36 AM)ObscureCoder Wrote: [\[CITE\]](#) Apart from how some of the code could be adjusted this is a VERY good tutorial. I recommend it to anyone starting out.

After someone covers this they should cover PSD to responsive HTML & CSS.

OP - Looks like you put a ton of effort into this, well done.

You're using an outdated part of HTML though; tables. No-one uses them on sites anymore, it's floating divs now.

I know you said they are not being used for design but they shouldn't be used for anything really.

tl;dr - You said you used questionable standards.

I still stand by the use of tables in the proper manner. Tables are the most efficient way to display data.

I thanks you for your reply

(May 31, 2014 07:31 AM)Argentium Wrote: [\[CITE\]](#) <http://prntscr.com/3obvkv>

If you have time, you can also explain the attributes for each tag. Nice tutorial, but more-of 'tl;dr' imo (no offence). I noticed some sentences which made no sense, will point it out later onwards, very sleep now. Good job! ?

There were several places like that, will go back and fix them. Slightly rushed post as I don't have access to anything but a tablet today.

Thanks for the reply.

\$lick - June 12, 2014, 10:40 pm

Sad to see this HQ thread die so fast.

Thanks for the tutorial!

SIM-1676753 - June 12, 2014, 11:13 pm

Looks like a very nice tutorial. I really hope this gets more recognition.

\$lick - June 12, 2014, 11:25 pm

(Jun 12, 2014 11:13 PM)Mr. Trvp Wrote: [\[CITE\]](#) Looks like a very nice tutorial. I really hope this gets more recognition.

Do you just fucking follow me around?

Stahp, You're scaring me.

SIM-1676753 - June 12, 2014, 11:38 pm

(Jun 12, 2014 11:25 PM)?Lazy™ Wrote: [\[CITE\]](#) Do you just fucking follow me around?

Stahp, You're scaring me.

No, I legit like this thread. Why do you think I keep stalking you.

michael1026 - June 12, 2014, 11:42 pm

Wow, this is incredible. I've actually learned from this, even though I thought I knew a lot about HTML already. Well done.

reama - June 12, 2014, 11:59 pm

Nice tutorial, I will be using this a lot. Very HQ.

lol - June 13, 2014, 6:57 am

Thanks for this tutorial. Just copied the thread link and sent to my friend! He'll love this. Thanks again.

.vemla - June 13, 2014, 7:05 am

Well done, buddy.

This will help a lot of people out there.

Thanks.

Peter DragonSlayer - June 13, 2014, 7:28 am

Thats a great tutorial my friend!

Thanks for putting time in it. A noob like me can really make use of this, TY

Roob - June 13, 2014, 3:31 pm

really nice guide, specially the business part, congrats!

Eeluminati - June 14, 2014, 7:13 am

One of the best tutorials I have ever read on this site.

Big kudos.

Exemplification - June 15, 2014, 7:39 am

Have fixed some minor issues with spelling, typos, and understanding.

I thank everyone for their replies to this thread. Again, I'm open to any suggestions.

Exemplification - June 17, 2014, 9:28 pm

THREAD AMENDMENT

Linking to Files

Let's learn about linking files. The subject is not complicated but seems to give some beginners issues.

There are two different ways of linking to a file. You can either directly link to it with a web address, or you can link to it in relation to the current file.

Direct linking is fairly simple with the use of a web address such as "https://hackforums.net/". Using this within the <a> element will look like the following

Code HackforumsOnto linking to other local files

Below is a sample file tree with / defining the root. //directory1/index.html

/directory1_1/double.html

/directory2/dir2.php

rootfile.html

Let's say we are on index.html and want to link to rootfile.html we can do this by preceding the file location with a forward slash, like follows

Code Root FileThe preceding forward slash (/) starts in the webroot, or the highest directory level available to the file.

Now, what if we are on double.html and want to link to index.html? We can do this by going up a level. Moving up a level is denoted with "../" An example of this use is following:

Code index

Now, one last option. Let's say we are on index.html and want to link to double.html. We can do this by providing the path relative to the current file (index.html). An example follows

Code double

Exemplification - June 23, 2014, 6:55 pm

THREAD AMENDMENT

Getting Your First Clients

Your first clients seem to be the hardest to find. You're new to the world, don't have much real world experience, and don't have much references to give. Let's go over some basic tips to make this stage of your freelancing less painful and less brutal.

Where to Look

Unlike a department store, your customers do not always come to you (rarely do they ever). It is your responsibility to find clients and offer them something they need or should have to make their life and business more successful.

Local Community

There is something about people in these recent days that think that local connections are not worth anything. These people are completely wrong. Your best long term business will most likely come from some clients in your local community. That pizza store on Main? Give them a website, add a loyalty program they can manage, and make their life easier (along with their customers). What about the boot store? Give them a website that looks good and gives basic information, but also add small features that will bring them business. Add a boot size tool, blog, tips, or whatever. If they make money, you will make money. Don't be afraid to go talk to the owner either unannounced (during an off time in business), or call them and set up a day to come in and talk to them.

This is also a point in your career that you should have some business cards. Nothing fancy, just a hundred or so of them with your contact information on them. Give each business owner a few of them and they just might pass it on to someone else.

These local businesses will stay with you if you provide them what didn't know they needed. Continue to innovate for them and you will have lifetime clients.

Local, but not so local

The local community is nice, but can sometimes be hard to jump into due to the social trends that seem to take over small towns or subgroups in a city. Extend your reach. Use services like Craigslist, backpage, Ebay Local, or even the local news paper. While these will probably not get you lifetime clients they will give you both experience and projects to show off.

Every time I mention to someone using Craigslist or Backpage they always seem to frown. While their reputation is not always five stars, gigs can be found on them and many times are reasonable pay. Using these sites will grab you clients that may be a bit away from your local community and can be a lifesaver if you are desperate for clients. Be careful with these clients; place all orders under a legal agreement and ensure that the person you are making the deal with actually is who they say they are. I've been down the path of having someone cheat me out of services once and getting repayment for these services took ~90% of the profit to pay for legal fees.

Distance Freelancing

Like a personal long distance relationship, a freelancing agreement outside of the local community and areas requires both parties to put forth the extra effort to convey what each of them want and can provide.

Distance gigs can be found on sites such as guru.com, freelancer.com, elance.com, or odesk.com. Some of these sites require payment for full functionality.

I Am Ready To Contact A Client

Great, now begins the fun!

If possible always make initial contact by phone. This not only allows you and the client to be able to instantly cover any issues in your agreement, it allows each of you to kind of form a business relationship that seems to be harder to do if neither company, group, or business has a voice.

The beginning stage of your contact consists of at least two steps but can be many. .

Initial Contact (Consultation)

Bid for work

Changes to work, start step two again

The first step should have the objective of finding the desires of the client, offering input on the desires, and providing any answers to questions the client has.

The second and third step is your bid. Take a bit to come up with some basic plans for the site, and assign a reasonable rate to each one of them. Have additional plans with additions to offer too. Contact the client and give them your offers. Some will tell you right then, some will make you wait a week while they get other offers. If your offer is accepted, great! Get to work. If the client changes some points you need to remake your bids and offer them again.

Congratulations, you've gotten your first client. It will be the beginning of a long and successful road.

Bag - June 23, 2014, 7:53 pm

Probably one of the most HQ tutorials on HF!

A lot of great quality work here!

Also coming from a freelance background I agree with that section 100%!

If you guys haven't read this seriously do! This is the type of content this section needs more of!

As for recommendation I would love to see this entire thread formatted really pretty into a psd file and posted that way!

If I get a free weekend I just might do it for you. It would help begginers beter visualize alot of the great cod snippets here better!

Exemplification - June 23, 2014, 8:35 pm

(Jun 23, 2014 07:53 PM)Bo?s Wrote: [\[CITE\]](#) Probably one of the most HQ tutorials on HF!

A lot of great quality work here!

Also coming from a freelance background I agree with that section 100%!

If you guys haven't read this seriously do! This is the type of content this section needs more of!

As for recommendation I would love to see this entire thread formatted really pretty into a psd file and posted that way!

If I get a free weekend I just might do it for you. It would help begginers beter visualize alot of the great cod snippets here better!

I thought about doing an image for everything, but it would make it much harder to edit and add additional information. It would also depend on an external site to host images, it goes down the thread goes down.

Bag - June 23, 2014, 8:46 pm

(Jun 23, 2014 08:35 PM)Exemplification Wrote: [\[CITE\]](#) I thought about doing an image for everything, but it would make it much harder to edit and add additional information. It would also depend on an external site to host images, it goes down the thread goes down.

True!

I think more of just an image in addition to the thread!

Like all current info on an easy to read image or pdf with nice code and syntax highlighting!

That way you can maintain the main thread but if someone so chooses they could view the image too.

Off Topic Note: It would be awesome to have a image editor like google docs where select people could update and help maintain the image version of the thread!

Just thinking of ways I could help out or that would be helpful to noobs as this is such a great thread!

JumpWar - June 24, 2014, 1:53 am

Very HQ Tutorial great you had the time to put this together.

F-Nuke - June 24, 2014, 2:07 am

Seriously, I loved the Freelancing section. Great job mate.

Sommen - July 18, 2014, 7:02 am

why so few resposnes to this thread? it's just as HQ as OP. damn, thanks a ton!

.Rez - July 21, 2014, 2:49 pm

Well this tutorial really made some things brighter now thanks for the tutorial op

GhostSniper - July 28, 2014, 12:06 am

Something I created following your tutorial op.

Spoiler ([Click to View](#))

Was gonna add more before posting this but its 3am and I'm exhausted. Thanks for the tutorial, I learned a lot!

Muzzaa - July 28, 2014, 6:36 pm

Awesome tutorial bro, I enjoyed the read, I enjoyed reading these when I was learning etc, very helpful and HQ thread

Exemplification - July 28, 2014, 9:00 pm

(Jul 21, 2014 02:49 PM).Rez Wrote: ([CITE](#)) Well this tutorial really made some things brighter now thanks for the tutorial opThanks for the feedback!

(Jul 28, 2014 12:06 AM)GhostSniper Wrote: ([CITE](#)) Something I created following your tutorial op. Spoiler ([Click to View](#))

Was gonna add more before posting this but its 3am and I'm exhausted. Thanks for the tutorial, I learned a lot!Thats awesome, I'm glad it was of use to you!

(Jul 28, 2014 06:36 PM)BeeMuzza Wrote: ([CITE](#)) Awesome tutorial bro, I enjoyed the read, I enjoyed reading these when I was learning etc, very helpful and HQ threadThanks for the feedback!

Hash Slinging Slasher - July 28, 2014, 9:04 pm

I already knew the languages so I decided to have a quick scroll. All the content seems pretty exact. Nice thread OP.

Caillou - July 28, 2014, 9:14 pm

Very informative and high quality guide.

Will surely help people wanting to pursue web development.

markeloff - July 28, 2014, 9:15 pm

Awesome guide mate,

Helped me a lot, Thanks.

Exemplification - December 4, 2014, 9:02 pm

Some minor adjustments have been made correcting a few issues.

hello im me - December 6, 2014, 5:22 am

WOW! What an amazing thread will defiantly use this alot, thanks for the share! ?

black initiate - January 11, 2015, 10:22 pm

I just happened to stumble upon this page via the link in your signature. As a electrical engineer looking to branch out into the area of web development I figured I would give it a read. Your tutorial was both informative and practical while providing a nice starting point. Thanks.

jazzycool - March 19, 2015, 11:00 am

Amazing tutorial bro, keep it up, very detailed and easy to follow tutorial.

Nozemi™ - March 19, 2015, 12:38 pm

This tutorial is very well written! I enjoyed the part about freelancing, as I'm more or less freelancing. Currently not usually taking jobs, because I already have a full time job. Anyhow, very useful information. I'm quite sure everything else is very useful information to people who don't know this.

Septunx - March 19, 2015, 12:51 pm

Great tutorial, helped rewrite my code so it easier to read and also improved the layout of the website I am currently working on.

Mavspammer - March 19, 2015, 10:04 pm

Well I dont know what to say cos its fucking awesome, taking time to do this simply means you are very devoted

IpSiety - March 23, 2015, 10:24 pm

Bumping this because this is an excellent thread.

Nozemi™ - March 24, 2015, 1:11 pm

(Mar 23, 2015 10:24 PM)Increase Wrote: [\[CITE\]](#) Bumping this because this is an excellent thread. Yeah, this should be a sticky thread. Probably one of the most informative threads in this section, if not the most informative thread.

vendettalogzz - March 26, 2015, 9:21 am

I just completed HTML&CSS, been searching for threads about both to keep my knowledge fresh. Without saying much, this thread is DOPE

jbyrne49573 - March 27, 2015, 7:29 am

Look on Google mate wil get loads of help from people on there

Comedic - March 27, 2015, 9:06 pm

This tutorial is longer than mankind.

Very well written though, mate. ?

KarateMonkey - March 27, 2015, 9:42 pm

Amazing TuT! I just came here for the freelancer part but read all of it...

Rushax - June 16, 2015, 5:48 pm

Great tutorial, really nice read.

Read it solely for the freelancing purposes/cross-browser comparability but picked up a few new things I previously didn't know about css resets!

If you're adding anymore to this may I suggest:

- colour usage in 3 ways, (hex, pre-defined names & rgb) as this baffled me when I saw hex starting out

- use of tags

- absolute, relative & fixed positioning in CSS

- HTML5 W3C Validato

Overall though I think this really useful to new web designers!

8bits - June 16, 2015, 8:41 pm

With the freelance pro:con thing, I suggest to try your damn hardest to get hired.

It's hard as hell to get actual profitable income, at-least enough to pay rent, freelancing. Along with that is reporting IRS can get hectic. Altogether, I would only recommend freelancing to build your portfolio and grow your skills.