# Pro HTML

# Pro HTML

HTML 4.01
HTML5
HTML Living Standard

Martin Rinehart

# Frontend Engineering I: Pro HTML

Knowit Publishing, LLC

Hudson Valley, New York, USA

*Written and Designed in LibreOffice Writer, Published to PDF, all in the USA.*

*Printed as near to our customers as our technology permits.*

# Table of Contents

(Full TOC, 177 pages, available on Amazon.)

# Introduction

Why HTML?

HTML is the *lingua franca* of the World-wide Web. Would you like to put content online? You will want to mark it up with HTML.

HTML is not just the language used to mark up pages. Backend tools, such as Java Server Pages and the PHP language (the language of Wikipedia and, until recently, Facebook) send content to your browser. The content they send? HTML.

The client end of the Internet, formerly browsers running on computers, mostly desktop PCs, is exploding. Smartphones and tablets are browsing and running web applications. Almost all these applications are presented, whatever the device size, in one language: HTML.

We're not sure what you would like to put on the web. But we are sure what you need to learn first, to put anything on the web: HTML.

Why Pro HTML?

Writing professional HTML takes a bit more savvy but no more effort than writing amateur HTML.

Ready to learn? Let's go. It's not hard. (And, for those with allergies, it has no math.)

# 1 Write Your First HTML

This is a “knowit” (knowledge unit). That means (in this case) that it has an online part, as well as this print part. The online part is at:

MartinRinehart.com

(No need to type “www.”.)

```
Online: Knowits > HTML I > Welcome
```

That box labeled “Online:” (we call it the “drilldown”) shows you how to navigate when you get to the website. You click “Knowits” and another menu appears. You click

“HTML I” and you get to the *HTML* menu. You click “Welcome” and … Well, you probably guessed.

In this chapter you'll write your first HTML and see your first page in your browsers. It's not hard.

Right now, go to:

Online: Knowits > HTML I > Welcome

(You'll want to arrange a comfortable position where you can look at the screen and at this print. If your “print” is a PDF, printing it would be a good idea.) Start with the online explanation of a “knowit.” We'll wait.

## Knowit Recap

A knowit, knowledge unit, can combine media. This one combines print and online. Print is easier to read. (There are about a quarter million “pixels” in a square centimeter of this print. There are about 800 pixels in a square centimeter of a typical monitor.)  But putting your mouse on the printed page and clicking is not going to get you anywhere. Both media have advantages, so we use both.

Now, off to the Companion page, the start of the knowit for this chapter:

Online: Knowits > HTML I > Online > 1

(We hope you noticed that if “Knowits” and “HTML I” were already punched, it took just two clicks to get to “Online” and then “1”. The great goal of all frontend engineering: make the visitors job simple!)

Now, let's get started. Each chapter, including this one, begins with a brief “history.” We start from the dawn of time but we'll get to electricity (chapter 2), very quickly. (We're big fans of history. It's a lot of fun if you don't leave it to historians.)

## History—Before Electricity

Our universe (and therefore, our HTML) began with a big bang, about 15 billion years ago. Stuff went flying in all directions. Galaxies spun around. Stars congealed. Our Earth started revolving and orbiting our Sun.

Life began. Living things became plants and animals, at least one of which crawled out of the ooze to live on dry land. We got dinosaurs, then mammals, monkeys and a couple weird animals that walked on two feet: *homo erectus* and then *homo sapiens*.

If we measure this in millions of years (let's call them *megayears*), 14,999 were gone before the latest started. (15,000 will visit us again.) We'll divide the final megayear into two eras, pre-electric and post-electric. In the first era, *homo sapiens* hunted and gathered. Then he (or, perhaps more precisely, she) learned to grow crops and we got villages, cities, wars and empires. This gave us stones, bronze and iron; science and mathematics; and we began to fiddle with a strange form of energy, called "electricity."

We'll get back to this tale (with a very soggy Ben Franklin flying his kite) in Chapter 2. First, though, let's write some HTML!

http://www.umich.edu/~gs265/bigbang.htm, Age of  the universe: 4.6 x 10^17 seconds. If your calculator's not handy, that's about 1.5 x 10^11 years.

(Remember: you can click on that link online.)

# Websites—SEO

After History, we return to websites and specifically, to the planning and design work that precedes writing the first bit of HTML. For the opening chapter we abandon our normally organized approach to begin with SEO, Search Engine Optimization, the most important part of website marketing.

Marketing? Yes, marketing!

You want your website to have visitors, even if it's not commercial. If you are not prepared to pay big bucks, you want users to find your site when they search for one like it, and preferably to have your site first in their first page of search results. (Studies have found that almost no one looks at the second page of search results. Very few even look past the top half of the first page. Being number one is huge!) Search Engine Optimization, SEO, is the science and art of doing things that will get your site a top placement on the search engine results pages (SERPs).

As we discuss the various HTML tags we'll highlight those that are really important for SEO. In this chapter, we'll cover, among others, the `<h1>` tag. It is used for the largest (and most important) heading. It is vital for SEO. If someone searches for “whole grain cereal” and your site is about whole grain cereal you want to be sure that “whole grain cereal” (which is called, in SEO, a “keyword” even though it's a phrase) appears in your `<h1>` heading. Try diving into this article in the Engineers' track online:

Online: Engineers > Frontend Team > Fe SEO

If you ever expect to create the next Facebook, SEO is vital. If you want to create a good site for your hometown pizza parlor, or your friend just starting her law practice, SEO is vital. If you are an international spy and you want to have a site that isn't listed by any search engine, good SEO can help with that goal, too. Keep your antennae tuned for every mention of SEO here and online.

## Tools—Editor, Browsers

No craftsman should be without a collection of tools. If you're serious about your craft you want the very best tools available. If you want to create HTML, your very excellent tools will all be free. (You can pay more, if you insist. Some of the higher priced tools actually may have advantages. Photo editing is commonly done with Adobe tools, like Photoshop, which are very expensive. Very!) HTML tools are low-priced, if they are priced at all.

Your HTML work will be very easy to understand (if not so easy to do). You use a text editor to enter your HTML and then you use your browsers to look at the result. Let's start with a text editor.

### Notepad++

On Windows, Notepad++ is an excellent programmer's editor. If the file you are editing ends with the `.html` extension (and your HTML files will) Notepad++ assumes that you are editing HTML and configures itself appropriately. In fact, Notepad++ is so rich in features but still so easy to use that we devote a separate section in each chapter to highlight one of its “secrets.”

On a Mac? TextEdit, which comes with OS/X, is an excellent programmer's editor so there's no need to install anything else, but there is a reason to worry. HTML in different browsers is not quite standard. You'll need to test the various Windows browsers, which means you'll need to test on Windows. This is not a “test when you think you're done” sort of thing. It's a “test continuously, every step” to catch issues as you go along. For that reason, few HTML authors use a Mac (even though many may prefer a Mac for everything else). So if you don't have access to a Windows machine, try to arrange access to a good friend with one so you can test your work every day.

On Windows you could actually use Notepad, but you'll be kicking yourself for doing it. We'll show you, as we go along building our site, how a couple extra mouse clicks can edit several files at once. So visit

notepad-plus-plus.org

and download your Notepad++. Follow the instructions and you'll be ready to start writing HTML, which is what we do next. (By the by, the Notepad++ installer will ask you if you really want to use the old-fashioned, too-large icon on your desktop. We're still partial to that one.) And for you hopeless cynics, we use Notepad++ every day writing HTML and several other languages and get no other compensation for recommending it to you.

## Browsers

With Notepad++ ready, you've got all you need to write HTML. (We'll introduce you to other tools as we go along. With Lorem Ipsum you'll be writing—sort of—good Latin!) Now you need to test your work in browsers. No, that's not “a browser,” it's “browsers,” plural. We recommend Google Chrome, Mozilla Firefox, Microsoft Internet Explorer, and Opera. (You can test Safari in lieu of Chrome if you prefer. They're both built on the Webkit engine, so they're very much the same when it comes to rendering HTML. “Rendering” in cooking refers to cooking off excess fat—on the web it refers to displaying the text and graphics specified by HTML on a screen or other device.)

Download any of these browsers you don't already have. (Remember, the online part of the knowit is clickable. You'll find the download links online.) Get each installed and running. If you have the monitor space, make the “Refresh” button of each available for clicking. At least organize yourself so you can “Alt+Tab” your way through the chain to pick a browser.

## Hello, World!

Now let's put these tools to use. The first challenge you face using any new language (that means “new to you,” not new to the world) is to make your computer correctly run a program that says “Hello, World!”. So fire up Notepad++ and type “Hello, World!” on the top line. (No, there isn't really any HTML going on. Be patient.)

Save your “program” in a file named `hw.html` and file this in a folder that is quite convenient. (The name preceding `.html` is really up to you. The `.html` is almost mandatory. Back in the very old days, just after St. George slew the dragon, Windows was limited to three characters in the suffix, so `.htm` was also enabled. Don't use this unless you have armor and a broadsword in your closet.)

Now, in each browser, in turn, press Ctrl+O (or click File/Open) and select the file you just saved. Each of your browsers should now report “Hello, World!” in the top-left corner of the browser's window, which is called the “client area” of the browser.

Congratulations. You have just created and viewed a web page. No, there wasn't any HTML but you did use the two critical tools: text editor and browsers. Now let's get on to HTML.

## Head

There are two parts to almost every HTML page: the head and the body. Each chapter here will have one major section, often a short one, devoted to the head section's content and two major sections, usually longer, devoted to block content and inline content, the two classic types of body content.  In this chapter we'll actually begin at the very beginning, which comes at the beginning of your HTML page, just above the head section. Ready to write your first real HTML?

### Beginning HTML

Let's do it! Let's add some real HTML. One of the things that separates the professionals from the amateurs is the presence of a “doctype” declaration. These used to be quite complex bits of SGML, Standard Generalized Markup Language, on which HTML was based. Fortunately, HTML5 introduced a very simple doctype, just as effective, and all browsers support it:

```
<!DOCTYPE html>
```

A note on capitalization: HTML doesn't care how you capitalize its tags and attribute names. The XHTML dialect insisted on lowercase. As lowercase is easier to type, the XHTML convention was adopted happily. We capitalize “DOCTYPE” (as you see above) for reasons that are explained in an obscure document linked in the Chapter 9 Companion page. You'll put “DOCTYPE” in a template where you'll never have to type it a second time.

That doctype declaration must be the very first thing (top line, far left) in your HTML file. We'll explain this next. For now, add it to your file, preceding the “Hello” text, and test it in at least one browser. If you haven't made a mistake, you'll see no difference.

Online: Knowits > HTML I > Online > 1 > 1a

When it is time to try something online we'll have an example in the online portion of this knowit. You can look at the example if you need help. The drilldown tells you that you should be doing something online.

If you haven't done so, yet, back up to `<!DOCTYPE html>` and add the doctype to your `hw.html` page. View the page in a browser or two. Look at Example 1a in another browser. Got it?

Now arrange your browsers in some order that you like. You want to cycle through them in order. (Ours are alphabetical.) The idea is that you will continuously test in each of the four so that if issues crop up (and they will!) you are never very far from the cause of a problem.

## Why a Doctype?

In the bad old days, all browsers were different. All browser vendors used a different subset of HTML, added a proprietary collection of extensions and hoped that you would write HTML specifically for their browsers and not for the competitors. This became unbearable back at the end of the last century.

So all browser vendors started working together (more or less) in standards committees to create a standard HTML. (Our History lessons will cover this in more detail.) And they invented modes: “standards” modes (in which they were all supposed to operate the same way on standard HTML) and “quirks” modes (which was whatever way they had been going on their own, in the bad old days). Adding a doctype really is a directive to the browser: “I'm writing in standard HTML, so behave yourself! No quirks allowed.”

We will show you how to create a template from which you will start all your HTML files. The template will start with a doctype, of course, so you will never actually create any quirks mode pages. (Unless, of course, you are curious. Quirks mode does not make your computer start to emit smoke. At least we hope it doesn't. We haven't tried in years.)

## Blocks

Now, on to our first block tags. We'll cover tag basics here, then four block tags divided into just three sections (as the last two are inseparable).

A tag is the basic HTML construct. To start your first and most important heading, for instance, you use the `<h1>` tag, this way:

```
<h1>This Is the Most Important
Heading</h1>
```

More on the `<h1>` heading in just a bit. For now, we'll examine tags in general. All tags are enclosed in angle brackets. (They are just above the comma and period on a standard western keyboard. Get used to them as you'll type lots of angle brackets.)

Most tags come in pairs, one opening and the other closing. The opening tag suggests its type. The most common are the most terse. `<h1>`, for example, is a very terse way of saying “the most important heading.” The closing tag is identical to the opening except that it inserts a forward slash just inside the opening angle bracket: the `<h1>` starts the heading, the `</h1>` ends it. Now on to our three block groups, after a word about the terminology.

HTML5, including the Living Standard, does not use the terms "block" and "inline." It distinguishes between "flow" content and "phrasing" content. These are (roughly) the same as our block and inline content. The newer terms will mislead you in older browsers, such as MSIE 8. The older terms still work in all browsers, so we'll stick with them.

## `<h1>`**Most Important**`</h1>`

The `<h1>` tag pair surrounds your most important heading(s). Time to try one?

Online: Knowits > HTML I > Online > 1 > 1b

That was the drilldown. You said to yourself, "Self! Time to try this online!" And you did so. Right? (Warning: reading these words without doing the online work is really a waste of time. Would you learn to play tennis without swinging a racket and hitting a ball?)

When you write an `<h1>` heading, you are writing the item that will attract your visitor's immediate attention. Choose your words carefully. When you follow our examples in your learning files, be creative! If the lesson is on `<h1>` headings, use `<h1>` tags. But have some fun with the heading you put inside the tags. Something like:

`<h1>`I'm Writing HTML! I Rule!`</h1>`

SEO, real sites: Your visitor might be Google's web crawler, looking to index your site for inclusion in search results. Google has taught its crawler that words inside

`<h1>` headings are very important. That means they are VERY IMPORTANT! Make sure any keyword terms or phrases are included in your headings, especially your `<h1>` headings.

## `<p>`Paragraphs in paragraph tags.`</p>`

Time to write a poem! Enter something like this into your page:

```
Hear we are learning
HTML.
We'll master the skill,
And it will serve us well.
```

Try it!

Online: Online: Knowits > HTML I > Online > 1 > 1c

When you view your page at this point, your browsers stretch that poem out in a single line (maybe two if it doesn't fit in one). This is not at all what you want. Suppose you had done better than our doggerel. Maybe you started:

```
Whose woods these are I think I know.
```

Well, you certainly wouldn't want that line ruined, would you? (Robert Frost, “Stopping by Woods on a Snowy Evening”.)

What is happening here is that the browser is taking the liberty of formatting your words to suit the available space. It assumes that text will fill the width available. So it unwraps your poem. Browsers are smart enough, however, to leave blank lines between paragraphs. Try the improvements shown here:

Online: Knowits > HTML I > Online > 1 > 1d

With that improvement, you see two lines, as if you had written two prose paragraphs. This is closer to what you want. Be patient: we'll get it exactly right, soon enough.

## Unordered Lists

Now, a bit of HTML terminology. We have three kinds of lists that we can specify:

- Unordered (bullet) lists
- Ordered (numbered) lists
- Definition lists

We'll get all three. For now, we'll use the unordered (bullet) list, such as the one above. It takes two different block tags: `<ul>` and `<li>`. Both come with start and end tags. They stand for “unordered list” and “list item,” respectively. The list above, in HTML, would be created this way:

```
<ul>
     <li>Unordered (bullet) lists</li>
     <li>Ordered (numbered) lists</li>
     <li>Definition lists</li>
</ul>
```

This is our first example of nested tags. The `<ul>...</ul>` pair surrounds `<li>...</li>` pairs.

Ready to try your own? For the next example, think of a short list of things you like: pizza toppings, musical instruments, friends… Think of things you like and follow our example, but use your own list.

Online: Knowits > HTML I > Online > 1 > 1e

## Inline

A block, such as the four we've just covered, spans the full width of its container. Until you get advanced, its container is the page. If its content doesn't fit, it will cover as many lines as it needs. A new block will start below a preceding block. Inline content, by contrast, will run across the page, each new inline item will come to the right of the previous inline item until a line is filled. It will wrap to the next line when more space is needed.

These are directions for most western languages, of course. Our friends writing languages from the area in and near the Arabian peninsula—such as Arabic, Hebrew, Urdu and Persian—will be happy to know that their languages default to the reverse, right-to-left, direction. HTML also serves those who write the Asian languages that use word symbols often written from top-to-bottom. We will, however, focus on the western languages.

### Inserting Breaks with `<br>`

Suppose you just want to break a line, but don't want to separate it into paragraphs. You use the `<br>` tag. Here's the final version of our poem:

Online: Knowits > HTML I > Online > 1 > 1f

When you try that last improvement, you have what you wanted: a poem!

## Do Not Use the `<font>` Tag

Now we will introduce a tag you should not use. Bear with us. This will make sense.

As HTML has progressed, some original tags are “deprecated.” This means that their continued use is not recommended by the W3C standards body. Deprecated tags may be dropped from future versions of HTML. A particularly important change in recent years has been the introduction of CSS, the Cascading Style Sheets markup sub language. With CSS, the advice has become to use HTML for “semantic” markup and to use CSS for “presentational” markup.

The word “semantic” refers to the concept “meaning.” What does the language mean? Presentational markup specifies what the finished page should look like. WWW pages have a wide potential audience, and that audience includes visually impaired users who may not be able to see your page. For these users there are browsers that read web pages, others that present pages in braille and so on. Your presentational markup will be invisible to these people.

Are you going to go to extra trouble for these users? We hope so. For one reason, although the visually impaired are a small minority (beware of statistics to the contrary) there is almost no extra work required to support their special browsers. You just have to use CSS for your presentational markup. (The CSS alternatives to the `<font>` tag, for example, are far more capable. You'll want to use them if you can.)

SEO: And to be completely selfish, think about SEO. Remember that the Google web crawler, which will or won't include your site in search results, is not just “visually impaired,” it's completely blind! If you want visitors to come to your site you will be sure to make your site as accessible as possible to the visually impaired.

That said, there are two perfectly good reasons for using deprecated, presentational HTML tags. One is that you are making a site for your youth soccer team, an audience that is all fully sighted. By its nature, your site has no meaning to a wider audience. The second is that you are learning HTML. While you intend to get to CSS in due course, for now you want an interim way to display things on the web. For you, we introduce the deprecated `<font>` tag.

The `<font>` tag has three attributes, without which it is meaningless. An "attribute" of a tag is a named value, included within the tag. Suppose you wanted to have some text in a larger size. You could do that this way:

```
Make this <font size='+2'>text
larger</font>.
```

These are the three font attributes:

- `<font color='xxx' …>`
- `<font face='xxx' ... >`
- `<font size='xxx' … >`

(You can use as many attributes as you like. Want a large, blue font? Try `<font size='+2' color='blue'>` big, blue text here `</font>`.)

The color may be a number from 1 (smallest) through 7 (largest). The default size is 3. The size may also be a digit prefixed by a "+" or "-" sign to increase/decrease the size. The face may be a font family, like Arial or Helvetica, or a pseudo family, like "serif" or "monospace." For serious design typography use CSS; for HTML use one of "monospace" or "sans-serif". Color may be an HTML color specification. For font colors, stick to names like "red", "blue" and "black". We exit sites that use white text on dark backgrounds, as fast as possible.

Remember that serif fonts, such as this text, are most readable in print. Sans-serif fonts are more readable on computer screens. The serifs are those little squiggles that decorate the letter ends:

- serifs on 'g's and 'G's
- sans-serif 'g's and 'G's

With that said, take a look at our next example. We have some fun with our "poem." Have some fun with yours, too.

Online: Knowits > HTML I > Online > 1 > 1g

To do this work (you really don't want to do this with fonts, but we will teach you the technique) make one change at a time. Edit, then view. Edit, then view. Be sure you keep changing browsers while you do this. Now a word about fonts and typography.

Professional designers take college courses in typography. If that is not you, stick to the basics. The experts employed by the browser makers have chosen their most readable fonts (with one exception) for general use. Don't overrule them unless you know what you are doing.

The `<span>...</span>` tag pair establishes the generic inline page element. Like `<font>`s without attributes, it does nothing by itself. The online material shows several examples of `<span>`s combined with just a wee bit of CSS, to give you the general idea. No need for you to use `<span>`s yet. This is just FYI.

If you really like serif fonts (such as the one you are reading right now) use “Times New Roman” in lieu of the generic “serif” font family. (MSIE chooses a truly horrible serif font by default.)

Online: Knowits > HTML I > Online > 1 > 1h

## Project—Pick a Topic

This knowit is organized around your project, which gives you both a lot of freedom and a lot of responsibility. For starters, take a look at our sample:

Online: Knowits > HTML I > Project

Your assignment for this chapter is to pick a similar topic for your own project. Be sure you include a class of similar things (people are good), a geographic dispersion and a time component. These let you do an image map (such as our map of Europe where you can click on a scientist) and a time line (that will teach you all about HTML tables).

If you can research your project in Wikipedia you will have a ready source of images that are available for your use without danger of a nasty-gram from an irate attorney asking for damages for your copyright infringement. You are far safer covering historical figures than using living people.

## Np++ Secrets—Handling Tabs

While you can generally just use Notepad++ as you would any other text editor, it has lots of features that make programming easier. Many of them are hiding right in plain sight. We'll point out at least one in each chapter.

Many are best tried online, including the tab handling. (It depends on little symbols that would be very hard to show in print.) So follow this drilldown and try it for yourself.

Online: Engineers > Frontend Tools > Notepad++ > More Np++ > Tabs

## Character Entities

A standard keyboard can produce fewer than 100 characters by typing a key or pressing Shift and typing a key. An original IBM PC expanded the “alphabet” of symbols to a full 256 possibilities. (That's a full byte: eight consecutive bits.) This wasn't enough. We expanded again from one to two bytes per character, and HTML now has 64,000 possibilities, of which about 4,000 are regularly used.

So how do you get to all these characters without a keyboard the size of a soccer pitch? (That's roughly the size of an American football field, end zones included.) Character entities.

You can use numbers, both decimal and hexadecimal or, for a smaller but handier selection, you can use named character entities. We'll start with three that flag intellectual property for its owners (who may be particular about the uses of their property). These are:

- Copyright: `&copy;`
- Trademark: `&trade;`
- Registered Trademark: `&reg;`

Return to the chapter's Companion page for examples:

Online: Knowits > HTML I > Online > 1

A named character entity starts with an ampersand, “&”, then has a name and ends with a semicolon, “;”. Enter these into any handy web page to learn how to use them, to see them on a web page and to help you remember them.

## Quiz

Choose the word or phrase that best completes each sentence.

1) SEO is critical to
   a) optimize searching speed.
   b) use Google effectively.
   c) attract visitors to your site.

2) The most important browser for website development is
   a) Firefox.
   b) Microsoft Internet Explorer.
   c) Google Chrome.
   d) All of the above.

3) The two parts of most web pages are
   a) head and body.
   b) tags and attributes.
   c) doctype and language.

4) The `<p>` tag is
   a) a part separator.
   b) used for parenthetical remarks.
   c) for paragraphs.

5) The `<ul>` tag is
   a) an inline tag.
   b) a block tag.
   c) an inline tag with attributes.
   d) a block tag around `<li>` tags.

6) The `<font>` tag is
   a) used for all wisdom.
   b) a modern styling device.
   c) deprecated.

7) Recommended font styles and families include
   a) Times, Helvetica, and Courier.
   b) Times New Roman, monospace, and sans-serif.
   c) Times Roman, Arial, and Bookman.

8) The default starting page is
   a) `home.html`.
   b) `index.html`.
   c) `start.html`.

9) The `<br>` tag
   a) has no closing tag.
   b) must always be properly closed.
   c) requires attributes.

10) Character entities have
   a) names and octal numbers.
   b) octal and decimal numbers.
   c) names and binary numbers.
   d) names, decimal and hexadecimal numbers.

# 2 Looking Good

In this chapter we're going to start creating professional web pages. We'll start a template, from which you can start all your other web pages and your project's home page. Time to open the Companion page:

Before we get started, let's roll our history forward toward the age of computers.

## History—Before Computers

The last megayear (of about 15,000 since the Big Bang—see “History” in Chapter 1) leads us to the age of electricity.

The ancient Egyptians knew that some fish defended themselves by giving electric jolts. Static charges were always present for those who pet even the gentlest cat. One 18th century tinkerer, with more curiosity than common sense, flew a kite in a storm, demonstrating that those great flashes called lightning were, in fact, electric. We would finally learn the relationship between electricity and magnetism, which led quickly to the electric motor. (Franklin's was impractical. Faraday's was better.)

The age of electricity really got rolling when Thomas Edison started to find interesting uses, such as illumination, phonographs and moving pictures. (Our French friends are saying, “but Lumiere was first with electric light!” True, but in engineering, getting things to actually work—and that means work at an affordable cost—counts for a lot.) “What good is this electricity?” Edison was asked. He answered, “What good is a baby?” (Or was that Franklin, paraphrased by Faraday?)

We already had telegraphy and then telephony. We would get radios that could actually pull music right out of thin air! Vacuum tubes let a very small signal (like a needle in a phonograph) modulate a very large signal (like one driving a speaker). And then Turing figured out how we might use these tubes to actually perform computations and the computer was born. Schockley's team at Bell Labs figured out how to replace the vacuum tube with transistors and we were off to the races.

If we measured the last megayear of mankind in units of about three human generations (66 years, which we could call *grandpayears*), again we'd have 14,999 out of 15,000 gone. The last grandpa year gets very interesting!

Clickable links on your Companion page: a brilliant note re Ben Franklin and the IEEE official history of electricity.

## Websites—Design

Our histories are about to slow down. Our website building is about to speed up. In Chapter 1 we looked at SEO, because building a website with no visitors is a waste of time. Here we'll back up to the beginning and look at the design of your project website.

Many books have been written on website design. The ability to design a good-looking website is a wonderful skill, and one that is not easy to master. You won't learn it here.

If your budget permits, hire a designer! If your budget doesn't have a fat item for website design, you'll have to do it yourself. So here we present a few pointers.

Begin with another look at our sample project:

Online: Knowits > HTML I > Project

Note that every page looks a bit like every other page. They all share the same background color. They all have the same navigation bar at the top. The title and the double lines that separate navigation from title from page are all the same.

You want to choose some design elements for your site that are also all the same. And you don't want to labor over your choices at this point. You can change your mind as you get more into your project. (In fact, we'll show you how to use Notepad++ to edit every page in your site with a single replace command.)

Design software can be frightfully expensive. Alternatively, you can grab a few sheets of white paper from your printer and some felt-tip markers. (Few designers work in HTML. It's too slow.) Steal some crayons from your baby sister, if that's all you can find. Start designing.

Begin with navigation. A simple list of pages, such as we have chosen, is easy to implement in HTML, and it's easy for a visitor to understand. We'll get into this more deeply in Chapter 4. For now, keep it simple and keep it consistent. Navigation, for left-to-right reading languages, should be either at the top or down the left side. Going down the left side requires CSS. For HTML, stick to the top.

Page color? You can leave it unspecified. Your visitors have either left their browsers to show basic white, or have chosen their own colors. Choosing a site-specific color helps to tie your pages together, but don't stray too far from white.

Graphics? We hope your topic comes with a natural set of images, such as the portraits we've used for our scientists. Their painters all seemed partial to brown backgrounds, (which we thought about when we chose a page color).

Typefaces? Use "sans-serif" for your entire site and use "Times New Roman" for headings, just for variety. (We chose "Times New Roman" for the main text. It's not as easy to read online, but we didn't have a lot of reading material in the site. It seemed appropriate for the 17th century topic.) Ignore this advice if you are a designer who knows typography. For really enjoying the design of websites, books and even TV shows, your life will be richer if you learn a little bit about typography. But that's not this book's subject.

Now think about decorative elements. Keep them simple. A double-line rule between sections is all you'll really need if your material has a nice mix of print and images.

Finally, avoid the temptation to center things. The very worst designs on the web are the ones with everything centered.

## Tools—Opera

Why Opera? As a browser it has a tiny market share. We do almost nothing in Opera that's different from what we do in important browsers, such as Chrome and Firefox. There aren't many Opera-specific “gotchas” (and the few we've found are deep into the wonders of JavaScript programs, not HTML).

So why Opera? The Error Console.

HTML is very forgiving. If you make a mistake, browsers will simply ignore the mistake. (You meant to type `<h1>` but your right hand wasn't paying attention so you got `<j1>`.)  You'd notice that your heading wasn't heading-sized, perhaps, and look to find the trouble, but your browsers wouldn't say anything about it.

So turn on the Opera Error Console. If you leave just a couple centimeters showing to the left of Opera, you'll see when it reports any problem. One click gives you an explanation. (Mostly, it's just a simple typo. Tell your hands to pay attention!)

To turn it on, see the Chapter 2 Companion page. It has a big color picture. The keyboard shortcut is Ctrl+Shift+O. (We're not big fans of complex "shortcuts" but this one we know well.)

## Head—Commented Template

In this chapter's "Head" section we'll cover comments, notes to yourself, and the three basic blocks tags of almost every HTML page.

## Comments to Yourself

A comment is a note to yourself. The text marked as a comment is read by the computer only to find where it ends, after which the computer will get back to its business. You, on the other hand, are vitally interested in the comments because, after all, you took the time to write them. (If your work is inherited by someone else, the comments become that much more important. "What," he or she asks, "is going on here?")

A comment is formed this way:

```
    <!-- this is an HTML comment -->
```

The comment is commonly put on a line by itself. It may span multiple lines, or it may be on a line with your HTML content, too. The important thing is the start and end characters.

We always start our pages (just after the doctype) with a comment that tells you the file path and name, along with a few words describing the page. We end (at the very end) with a comment that tells you this is the end of the page:

```
<!DOCTYPE html>

<!-- folder/foo.html – page describing
“foo” -->


. . . page content here

<!-- end of foo.html -->
```

The Companion page link will tell you more, if you really want to know.

Now on to tags. One of them is actually the `<head>` tag.

### The `<html>` Tag

In Chapter 1 we entered and viewed HTML without the `<html>` tag. (And with no `<head>` nor `<body>` tags, either.) How did that work?

If you omit these tags, the browsers assume you are entering HTML in the `<body>` section. You can omit these tags, but if you omit them, HTML checkers, such as the W3C Validator we'll use in Chapter 7, will be annoyed. Putting these tags into a template, that we'll get to momentarily, gets them into your pages with no extra effort at all.

The `<html>` tag should start, and the `</html>` tag should end every HTML page, excepting only the doctype and comments.

### The `<head>` Tag

If you want to have head-section material in your HTML page (and you definitely will want this) it must come between `<head>` and `</head>` tags. Omit these and you will not have a head section and any head section tags will be ignored.

### The `<body>` Tag

The body section comes between `<body>` and `</body>` tags. Including them is necessary if you want to validate your pages.

### The Page Template

You'll see how these tags work together when you make your page template, per the directions at:

Online: Knowits > HTML I > Online > 2 > 2a

Save your `template.html` file where it will be handy, but not too handy. (It is very easy to ruin a template by typing page-specific HTML into a generic template where it will be totally wrong, the next time you want to use the template). And then save your template in another place, where it will be available if you (when you?) inadvertently ruin the original.

## Blocks—`<h2>` through `<h6>`

As you may have guessed, there are more headings than just `<h1>`. There are six, in total. As a practical matter, the first three or four have uses as headings in web pages. The last two, `<h5>` and `<h6>`, are only useful in places such as the fine print of legal documents. `<h4>` is very close to the size of regular text, though often emphasized (as with bold or italic attributes).

The second example shows you how to build your own menu of headings, so you can see for yourself. It is also the first example to use your new template file. Instructions are online.

Online: Knowits > HTML I > Online > 2 > 2b

## Inline—Mono and Global

We'll cover two topics in this section. First, we'll look at all the inline elements that use monospaced fonts.

Monospaced font elements are used for showing program code (for example, the file listings in the online pages). And they are used for showing old-fashioned keyboard input, screen samples and variable names. (If you guess that HTML was originally used to talk about computers and computer programs you are absolutely right.) Will you ever use these? It depends on your subjects.

Then we'll move on to the universal (or global) attributes that can be used with every tag, block or inline.

### Monospaced Font Tags

There are four monospaced font tags available. They are:

- `<code>` such as for code listings
- `<kbd>` for showing typed (keyboard) input
- `<samp>` for computer output samples
- `<var>` possibly for variable names

We'll show these in example 2c and move right along. If you can surround headings with heading tags, you can surround monospaced elements with any of these tags.

## Universal (or Global) Tag Attributes

You've seen `<font>` attributes such as `size` and `color`. Some attributes are called “universal” in HTML 4 and “global” in HTML5. Both mean the same thing: you can use them with any tag in HTML.

Neither “universal” nor “global” should be taken too literally. Some tags are quite unsuited for some attributes. (What color is a `<br>`? Perhaps the same color as one hand clapping?)

The attributes are used with any tags where they make even a little sense. However, we haven't discussed HTML 4 and HTML 5 standards yet. Our histories will treat them fully, but we're not there. For the moment, HTML 4 is the current standard HTML. It's been with us since 1999. HTML 5 is the coming standard. It's due to be made final in 2014. (It is less common for a standard to be finalized on time than for one to be years late. We'll see.) In the meantime, we have taken the attributes that are both universal in HTML 4 and global in HTML 5 as the set that you should know about and use today.

The dual-standard universal/global attributes are:

- `class`
- `dir`
- `lang`
- `style`
- `title`

The `dir` attribute refers to language direction. If you will be using Arabic and Hebrew (or other Semitic languages), you will need to look into this on your own.

The `style` attribute lets you use CSS styles on individual elements. This jumps ahead to Volume 2 of this series. For a taste, the following would make a paragraph's background red:

```
<p style='background-color: red'>This would be
red!</p>
```

The `class` attribute is also CSS-dependent. You would use it when you wanted a certain class of paragraphs to all be red, for instance.

Last, and most immediately helpful, the `title` attribute lets you assign a title to an element. When the user hovers the mouse over an element with a title, the title is shown as a tooltip.

The online example shows monospaced elements with tooltips:

Online: Knowits > HTML I > Online > 2 > 2c

## Project—Create Your Site

Now that you have a template, you can create the first page of your website. The first page is called `index.html`. In a browser, if you ask for "google.com" in the address bar, your browsers see that the address (more exactly, the URL —Uniform Resource Locator) has no page specified. So it adds `index.html` for you. The browser also sees that your address has no "transfer protocol" so it adds the default—http for HyperText Transfer Protocol—for you and navigates to:

http://MartinRinehart.com/index.html

You'll need the page name for now. Remember that home pages are called `index.html`. You'll need the transfer protocol when you start creating links to external pages. We'll remind you when we get there.

You'll also need a folder. Something like “english-lady-novelists” would be good (assuming, of course, that your project is about English lady novelists). This is a time for a word on folder and file names.

The Windows operating systems are “case-insensitive” for folder and file names. All of these are pathnames for the same Windows file:

- `foo/bar.html`
- `Foo/Bar.html`
- `foo/Bar.HTML`

In Unix-derived operating systems (including Linux and Apple's OS/X) those names are all different. Folders `foo` and `Foo` are different, separate folders. Files `bar.html` and `Bar.HTML` are different, separate files. Why do you care?

You care because you will develop on your local system (likely Windows) and then upload your finished website to an Internet Service Provider (ISP) when it is ready for the world at large. Your ISP will probably use Linux-based servers. Differences in case in the folder and file names that were hidden when you built your system suddenly are broken links (and, as Murphy's Law would predict, at the worst possible moment: just when you are ready to go live).

What do you do? The simple, foolproof solution is to never, ever use a capital letter in a folder or file name. There are similar “gotcha”s in multi-word names. We always separate our multi-word names with hyphens, not underscores or spaces, and recommend that you do, too.

Now, let's build that starting page. (Remember that when you look at our examples, they are all for our sample project. Your names should all be different, of course.) Start by making your folder. Then open your template and immediately save it as `index.html` in your new folder.

Add a major heading and you are done.

Online: Knowits > HTML I > Online > 2 > 2d

If you are smart, you will use the blank space to make notes to yourself about the pages you plan to add and whatever else comes to mind.

That was easy, wasn't it? This project work will be much easier than you might have guessed. In fact, so will HTML!

## Np++ Secrets—Vertical Space

Vertical space is a precious commodity if you create web content. Except in the very early stages of a project, your editor isn't tall enough to show your complete pages. This is odd.

The history of computers is full of constraints that have stopped being constraining. In the old days, computer memory was fantastically expensive and therefore scarce. Today a gigabyte of extra memory costs little more than a first-class sandwich. Disk space? It's nearly free if you don't make a habit of storing full-length movies on your hard drive.

But vertical space is still a limited, expensive commodity. Notepad++ can help, but only a little. So do check the following page, and do take its advice regarding the main menu (even if it scares you!).

Online: Engineers > Frontend Tools > Notepad++ > Lines

## Entities—Accents

The World-wide Web was originally based on U.S. Standards, such as ASCII: the American Standard Code for Information Interchange. The original ASCII used some control characters (linefeed, return and so forth), 26 letters (A through Z, uppercase only), digits zero through nine and some punctuation marks. All of these could be represented in just six bits. As communications lines were fabulously expensive, six-bit letters were chosen over 7-bit letters. (Seven bits would have permitted lowercase letters, too).

Official use of 8-character alphabets didn't take over the Web until 2007. Today's web is based on 16-character "letters" which allow for encoding most modern languages, including Asian ones (based on full-word symbols) as well as many now extinct languages important to scholars who study the development of written language.

In past years, and still today, character entities filled in the gaps for western languages with a simple, elegant system that permits a variety of accents that let you write, more or less correctly, most of the languages in use in Europe and the Americas. This is described on the Companion page for Chapter 2.

Online: Knowits > HTML I > Online > 2

Though we're not big fans of memorizing facts (in an era when "google" has become a verb—why memorize when you can google it?) but knowing some character entities is very handy. So these will be on the quiz (where you can still google the answers, but it will take you a little longer).

## Quiz

Choose the word or phrase that best completes each sentence.

1) An inventor, asked about the value of electricity, answered, “What good is a baby?” His name was
   a) Edison.
   b) Lumiere.
   c) Franklin.
   d) Farady.

2) Website design is
   a) an art best done by professional designers.
   b) worst when everything is centered.
   c) concerned with site navigation.
   d) all of the above.

3) The best font families for non-typographers are
   a) Arial and Helvetica.
   b) Times Roman and other traditional fonts.
   c) Times New Roman, sans-serif and monospace.
   d) Bookman and Vera Sans.

4) Three page structuring tags are
   a) `<html>`, `<doctype>` and `<body>.`
   b) `<header>`, `<body>` and `<footer>.`
   c) `<html>`, `<head>` and `<body>.`
   d) `<doctype>`, `<head>` and `<foot>.`

5) Heading tags `<h5>` and `<h6>` are
   a) used for mid-sized headings.
   b) used after the first four headings.
   c) too small for common use.

6) The home page is commonly named
   a) `home page`.
   b) `home.html`.
   c) `index.html`.
   d) `home-page.html`.

7) Folder and file names are
   a) not case-sensitive.
   b) case-sensitive.
   c) never used in lowercase.
   d) best in all lowercase.

8) Websites are most commonly served by
   a) Windows computers.
   b) Apple computers.
   c) Linux computers.

9) Modern HTML uses up to
   a) 6-bit characters.
   b) 7-bit characters.
   c) 8-bit characters.
   d) 16-bit characters.

10) Character entity accent names include
   a) acute, circumflex and tilde.
   b) grave, tilde and umlaut.
   c) acute, grave and umlaut.
   d) acute, tilde and uml.

# 3 Populate Your Pages

There are lots of important milestones in this chapter. To begin, our history will get to the birth of the Internet, an event as important as any in the last century. Then we'll plan the pages in our project website. Ready to use Latin in your developing pages? (Faux Latin, of course, and it's been used for centuries.) Ready to add colors and pictures (big things!) and footnotes (important things!) to your site?

Start online at the Companion page.

Online: Knowits > HTML I > Chapters > 3

Now, here comes the Internet.

## History—Birth of the Internet

By the middle of the last century, we had huge mainframe computers and atomic bombs. A chief source of research funding was the U.S. government, which thought that U.S. security depended on these things. The Defense Advanced Research Project Agency (DARPA) wanted a way to interconnect the computers (that controlled the bombs). They wanted this to be so secure that it would withstand a nuclear attack on the U.S.

A technology called “packet switching” was invented that let each computer in a network function as a controller of the network. If one fails, the rest of the network can continue operating. This answered the essential defense problem.

On October 29, 1969, a student programmer sent the first message over the “ARPANET”. It was the word “login”. The network crashed after two characters. Later the same evening, the whole word was successfully sent and one computer logged in to another. By December 5, four separate computers were communicating over the ARPANET and the Internet was born. (We're not magic-driven numerologists, but that was about 15,000 days ago.)

http://www.pbs.org/opb/nerds2.0.1/geek_glossary/packet_switching_flash.html Packet switching

## Websites—What Pages?

Website design only has a neat set of steps in theory. In practice, you have to do it almost all at once. You decide what it will look like and you decide what pages you have before you get started.

In this chapter you will add all the pages you want. Right now, begin thinking about what you want. In our sample project, that is:

- home page
- time line
- map
- one page per scientist
- Newton's “quotes”
- credits

Your design should be similar. If you want your “quotes” all in “Latin” the next section will show you how we do it.

## Tools—*Lorem Ipsum*

In the 1500s an unknown printer wrote these immortal words:

> *Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut quam arcu, sodales ac facilisis sed, tempus ac orci. In scelerisque ultrices fermentum. Curabitur quis mi ligula.*

What do they mean? Absolutely nothing. The printer was soliciting business. He wanted to show potential clients what their documents might look like. He'd seen some work by Cicero and he invented a way of dropping in letters that would look like Latin. Ever since, “Lorem ipsum” (the first two words of his classic text) has been used to demonstrate the look of the page, without the distraction of having real text.

Where do you get Lorem Ipsum? You google. Sites like `lipsum.com` (clickable link on Companion page) have generators for you. Most are free.

Here you have one of the least likely distinctions between professional and amateur HTML Professionals know where to go to get their nonsense.

## Head—`<style>`, `<script>`

This chapter's lesson on the head section will be brief.

You will see pages that interest you, not for their content but for their HTML. You will right-click the page in your browser and it will give you an option to “View Source” (or “View Page Source” or just “Source”). You will look at the page's HTML. Near the top, you'll see the head section.

In addition to head-specific tags (page author, keywords and so on) you may see a `<style>` section in the `<head>` section.

The `<style>` section is where you will see any page-specific CSS. (Larger websites will have most of the CSS in separate CSS files. Those can be applied to multiple pages or even the whole site.) The `<style>` section ends with an `</style>` tag, as you might have guessed. We hope this encourages you to go on to Volume 2 in this series. It's all about CSS.

Many pages also have `<script>` sections in the head, also ended by `</script>` tags. This is for JavaScript program code. Today it is not considered best practice to put JavaScript in the head. (Putting it at the end of the body is preferred.) But you will see it there in older pages, and in pages where the author isn't aware of current best practices. We hope the `<script>` encourages you to go on to Volume 3!

In the Companion page, we show these tags added to the template.

# Blocks—Colors, `<ol>`s

Now we dive into the content of the body section, beginning with block elements. In this chapter, we'll concentrate on the attributes that add color to blocks.

Today, CSS is recommended for color. Color is, of course, presentational, not semantic. It is used to get the appearance you want, such as a page color. Of course, you might use the color red to highlight warnings, which would be semantic. (If you do so, remember that some people cannot see the color. Use words, too.)

## The `bgcolor` Attribute

Prior to CSS, color was specified in the `bgcolor` (think “background color”) attribute of blocks. We'll get to examples momentarily.

The `bgcolor` attribute was deprecated in HTML 4 and dropped entirely from HTML 5. We definitely recommend using CSS, instead. However, if CSS is in your future while HTML is in your present, don't be afraid to use `bgcolor`. We guess that all browsers will still support it for many years.

## Use `bgcolor` with `<body>` and `<table>` Blocks

Would you like a nice light brown, parchment feel for your pages? In the `<body>` section, try:

```
<body bgcolor='#f8f4f0'>
```

(Don't fret over `f8f4f0`. It specifies a color and we'll explain how, below.) It's usual to think of blocks as elements you use in the body section. However, the body section is, itself, a block.

Would you like a nice light blue background for a table? Try:

```
<table bgcolor='#f0f0ff'>
```

We'll get to tables in Chapter 5. For now, remember that you can use the `bgcolor` attribute with tables, individual table rows and single cells within rows.

## Specific Colors

Now, how do you specify a color? You use names (like 'blue') or hexadecimal numbers. We're pretty sure you're familiar with decimal numbers; we'll explain hexadecimal numbers (“hex” for short). But first a word about computer colors.

Most computer displays use RGB colors. That stands for red, green and blue. A standard desktop monitor uses only these three colors, each in intensity varying from zero (off) through 255 (maximum on). Black is no color: R, G and B all turned off. White is all on: R, G and B guns all on maximum.

We use the names “white” and “black” as a shorthand for “all on” and “all off.” Otherwise, color names are nearly useless. Primary colors, “red” and “blue,” mean what they say: “red” is, for example, 100% red, no green and no blue. The name “green” stands for a green color, but not pure green. As colors, the primaries are suitable for decorating kids blocks, but not much else.

And that brings us to hexadecimal numbers. Decimal numbers use a “base” of ten, represented with digits as “10”. (Say that to yourself not as “ten” but as “one, zero”.) Two digit decimal numbers are one of `00, 01, 02, … 98, 99`.

Hexadecimal numbers use a base of sixteen, also represented with digits as “10” (again, “one, zero”). For single digits, hex numbers use the familiar “0” through “9” (zero through nine) followed by “a” through “f” (ten through fifteen). Hex is very convenient with computers which internally handle data as bytes: 8-bit chunks. In hex a byte is one of `00, 01, 02, … fe, ff`.

The hex numbering assigns two digits to each of the primary colors, preceded by a “#”: #`rrggbb`. The first two hex digits specify the amount of red. “`00`” means no red at all. “`ff`” means 255, the maximum. Ditto for the green and blue colors.

If you don't worry about numbers you'll be happy choosing your colors. It's like mixing paint out of the can. You pour a bit of blue into the white and stir. Too light? You pour in a bit more blue. Unlike mixing paint you do not need to wear old clothes. Most important, if you get a bit too much blue it's easy to take some out. (That's in HTML. It's not so easy with old-fashioned paint.)

Another very important difference between computer and paint colors is that adding dark colored paints gives you, if they're dark enough, a near-black. Computer colors are the reverse: add equal parts r, g and b and you get gray. Add more of each and you get a lighter gray. Add the maximum of each and you get white. The Companion page encourages you to experiment:

Online: Knowits > HTML I > Online > 3

At the risk of repeating, we'll say again: experiment.

But don't obsess! Some designers hang expensive color meters on their monitors. This made more sense designing for print than it does today. Today you could get the exact shade you wanted, only to find that it doesn't come out the same way in all browsers. Then you find that it's not the same on all monitors. And then you notice that colors change, and change a lot!, as you view your laptop from different angles. Have fun with colors, but don't obsess.

## Ordered Lists

Ordered lists, with one small addition, are just as simple as unordered lists. You just change `<ul> … </ul>` to `<ol> … </ol>` and your bullet list is now a “numbered” list. Now for that one small addition.

Unordered lists, as you recall, have a type attribute that can be one of:

- `type='circle'`
- `type='disc'`
- `type='square'`

Here's a simple, ordered list:

1. try an ordered list
2. try the type attribute
3. make an outline

By default, an ordered list is numbered. Numbered with simple, old-fashioned Arabic numerals. Now for the type attribute. It can be one of the following:

- `type='I'`
- `type='A'`
- `type= '1'`
- `type='i'`
- `type='a'`

Yes, those are the “numbering” styles you use with an outline. Your Companion page has samples:

Online: Knowits > HTML I > Online > 3

## Inline—Images

The National Center for Supercomputing Applications (NCSA, University of Illinois at Urbana-Champaign) released the Mosaic browser in 1993.It took the world by storm because it supported pictures. Until then, the World-Wide Web had been a text-only system. 1993 changed it forever. (Lead programmer Marc Andreesen, along with investor Jim Clark, went on to found Netscape and become the Internet's first mega-rich.)

Today, images are very simple. You just drop in an `<img>` tag, a few attributes and you have done what was, just 20 years ago an amazing feat.

The `<img>` tag has no closing tag. You don't put your picture into the HTML. You put your picture's address in the `src` attribute. Here's an example:

```
<img
    alt='Picture of my puppy.'
    border=0
    src='graphics/my_puppy.png'
    title='Picture of my puppy.'
>
```

(You could put the tag and its attributes all on a single line, if you want. HTML doesn't care. We like this format because it's easy for us to read, not because any browsers care.)

The `alt` attribute is the 'alternate' that will be read or otherwise used by browsers for the visually impaired. SEO! Remember that the `alt` attribute is the only way a web crawler can find out about what is in a picture.

Microsoft's browsers put a one-pixel border around images, by default. All other browsers default to no border. Most people do not want the browsers to add borders, but you have to specify `border=0` to keep MSIE from adding a border. Alternatively, you would have to add `border=1` if you wanted browsers to add a one-pixel border. This is so annoying!

The `src` attribute is the one that tells the browser where to go to get the picture. Many put pictures into a subfolder called "graphics," as assumed here. If your page is in `whatever/folder/page.html`, then the sample picture would be found in `whatever/folder/graphics/my_puppy.png`.

Last, we come to the tooltip. Some browsers will use the `alt` attribute for a tooltip, if you don't provide a `title`.

Some won't. If you include the `title` attribute, it will be used by all browsers as the tooltip. If you don't want a tooltip, use `title=''`.

Note that the slashes between folder, subfolder and file names go forward, Unix style, not backward, Windows style. Note also that there is no slash before the folder name in `src='subfolder/filename.xxx`'.

What kind of picture files? All common ones work well: `.gif`, `.png`, `.jpg` (aka `.jpeg`). Some less common ones also work well: `.tiff`, `.raw`, `.bmp`. Chances are that if you downloaded a file from the web or uploaded a file from your camera you can use it.

What size? The picture on your site will be whatever size the picture is in the file unless you specify the `height` and/or `width` attributes of the `<img>` tag. (If you specify one, not the other, the browser will maintain the original aspect ratio. If you specify both you can distort the picture.) It used to be a bad practice to use the browser to resize the picture (it was better to resize in expensive software like Photoshop). Today's browsers can resize quite nicely. Of course, if you start with a very small picture and display it at a very large size, it will be fuzzy. Browsers don't work miracles.

## Project—Your Pages

Here we'll create a starting page for each of the pages in your project website.

First, create a list of your pages. Give them readable names. `jane-austen.html` would be good for a page about Jane Austen.

Second, open your `index.html`. Save As … your second project page. (Your first is `index.html`, of course.) Now change the heading to correctly name your page. Repeat, but not before you read the next paragraph.

When you come to the people that will populate your site (or trees, or frogs or …) add an `<img>` for the photo or painting that illustrates that person. Write a sentence or two about each person. Or grab a little Lorem Ipsum. Visit our sample project, any scientist and right-click View Source if you have any question about how it's done.

Now, make one page for each of your project's final pages. Add images as appropriate. You now have the makings of a website. Or at least you would have if there were some way to navigate from page to page. We'll get there, next chapter.

## Npp Secrets—Sessions

A programmer's editor must, (not should, must!) store exactly what you type in its file. Anything else would probably be disastrous.

But Notepad++ may know more. The first, obvious thing is your position at the top, bottom or somewhere in the middle of the file. If you want Notepad++ to remember the extras, but not add anything to your HTML, you should know about its Sessions. If you want to have exactly three pages open for one part of your project, but have four other pages open for something else, you definitely want to know about Sessions.

Online: Engineers > Frontend Tools > Notepad++ > Sessions

## Entities—Three Footnotes

HTML lets you go as far as you like with footnotes. But if your needs are simple and three footnotes are enough, there is nothing quicker than the superscript character entities.

Online: Knowits > HTML I > Online > 3

If you want four or more footnotes, you can still use these for the first three.

## Quiz

Choose the word or phrase that best completes each sentence.

1) The Internet developed from a perceived need to
   a) communicate via email.
   b) hyperlink documents.
   c) survive a nuclear attack.

2) Website designs begin with
   a) the “look” of the site.
   b) the “look and feel” of the site.
   c) the content of pages.
   d) all of the above.

3) “Lorem ipsum” is
   a) the start of a famous essay by Cicero.
   b) Latin from the 1500s.
   c) non-Latin from the 1500s.
   d) a university slogan.

4) When used
   a) `<script>` goes in the head, `<style>` in the body.
   b) `<script>` and `<style>` go in the body.
   c) `<script>` and `<style>` go in the head.
   d) `<style>` goes in the head, `<script>` at the end of the body.

5) The `bgcolor` attribute is
   a) deprecated in HTML 5.
   b) used for image backgrounds.
   c) introduced in HTML 4.
   d) still available but not recommended.

6) The `border` attribute of an image tag
  a) is required if you don't want borders.
  b) is required if you do want borders.
  c) is required if you care about borders.
  d) is required if you don't care about borders.

7) The color black may be specified as
  a) `'black'`.
  b) `'#000000'`.
  c) both of the above.
  d) neither of the above.

8) The `alt` attribute of the `<img>` tag
  a) provides information to the visually impaired.
  b) is read by search engine web crawlers.
  c) may be used as a tooltip.
  d) all of the above.

9) Footnotes may be shown by character entities
  a) if you use four or fewer.
  b) if you adopt MLA style footnotes.
  c) if you specify the original source.
  d) none of the above.

10) Mosaic was
  a) the first text-mode browser.
  b) programmed at the University of Michigan.
  c) programmed at Michigan State University.
  d) commercialized by Netscape.