

www.agilityio.com

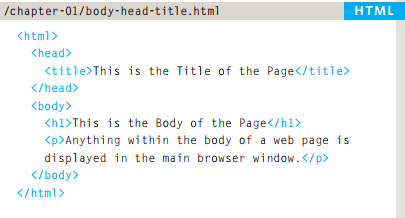
**HTML/CSS Training Note**

HTML Note

# Book “Design and Build Website”

## CHAPTER 1: STRUCTURE

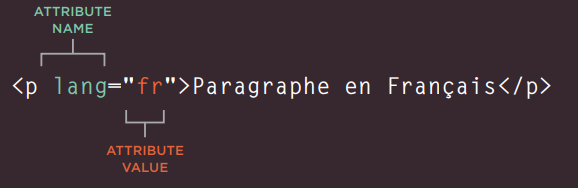
HTML pages are text documents.



The HTML code is made up of characters that live inside angled brackets — these are called HTML **elements**.

Elements are usually made up of two **tags**: an opening tag and a closing tag. (The closing tag has an extra forward slash in it.). Each HTML element tells the browser something about the information that sits between its opening and closing tags.

Opening tags can carry **attributes**, which tell us more about the content of that element. **Attributes** require a name and a value.



### BODY, HEAD & TITLE

<body></body> Everything inside this element is shown inside the main browser window.

<head></head> Before the <body> element you will often see a <head> element. This contains information about the page. You will usually find a <title> element inside the <head> element.

<title></title> The contents of the <title> element are either shown in the top of the browser, above where you usually type in the URL of the page you want to visit, or on the tab for that page.

## CHAPTER 2: TEXT

### HEADINGS

HTML has six "levels" of headings:

* <h1> is used for main headings
* <h2> is used for subheadings
* If there are further sections under the subheadings then the <h3> element is used, and so on..

### PARAGRAPHS

To create a paragraph, surround the words that make up the paragraph with an opening <p>

tag and closing </p> tag. By default, a browser will show each paragraph on a new line with some space between it and any subsequent paragraphs.

### BOLD & ITALIC

<b></b> make characters appear bold. The <b> element does not imply any additional meaning.

<i></i> make characters appear italic.

### SUPERSCRIPT & SUBSCRIPT

<sup></sup> The <sup> element is used to contain characters that should be superscript such as the suffixes of dates or mathematical concepts like raising a number to a power such as 22.

<sub></sub> The <sub> element is used to contain characters that should be subscript. It is commonly used with footnotes or chemical formulas such as H20.

### LINEBREAKS & HORIZONTAL RULES

<br /r> to add a line break inside the middle of a paragraph.

<hr /r> To create a break between themes — such as a change of topic in a book or a new scene in a play

**\*\*There are a few elements that do not have any words between an opening and closing tag. They are known as empty elements and they are written differently.**

### SEMANTIC MARKUP

There are some text elements that are not intended to affect the structure of your web pages, but they do add extra information to the pages — they are known as semantic markup

### STRONG & EMPHASIS

<strong></strong> The use of the <strong> element indicates that its content has strong importance. By default, browsers will show the contents of a <strong> element in bold

<em></em> The <em> element indicates emphasis that subtly changes the meaning of a sentence. By default browsers will show the contents of an <em> element in italic.

### QUOTATIONS

<blockquote></blockquote>The <blockquote> element is used for longer quotes that take up an entire paragraph. Note how the <p> element is still used inside the <blockquote> element.

<q></q> The <q> element is used for shorter quotes that sit within a paragraph.

### ABBREVIATIONS & ACRONYMS

If you use an abbreviation or an acronym, then the <abbr> element can be used. A title attribute on the opening tag is used to specify the full term.

In HTML 4 there was a separate <acronym> element for acronyms. To spell out the full form of the acronym, the title attribute was used (as with the <abbr> element above). HTML5 just uses the <abbr> element for both abbreviations and acronyms.

### CITATIONS & DEFINITIONS

<cite></cite> When you are referencing a piece of work such as a book, film or research paper, the <cite> element can be used to indicate where the citation is from.

<dfn></dfn> The <dfn> element is used to indicate the defining instance of a new term. Some browsers show the content of the <dfn> element in *italics*. Safari and Chrome do not change its appearance.

### AUTHOR DETAILS

<address></address> The <address> element has quite a specific use: to contain contact details for the author of the page

### CHANGES TO CONTENT

<ins></ins> The <ins> element can be used to show content that has been inserted into a document. The content of a <ins> element is usually underlined.

<del></del> The <del> element can show text that has been deleted from it. The content of a <del> element usually has a line through it.

<s></s> The <s> element indicates something that is no longer accurate or relevant (but that should not be deleted).Visually the content of an <s> element will usually be displayed with a line through the center.

## CHAPTER 3: LISTS

### ORDER LISTS

<ol></ol> The ordered list is created with the <ol> element.

<li></li> Each item in the list is placed between an opening <li> tag and a closing </li> tag. (The li stands for list item.)

### UNORDERED LISTS

<ul></ul> The unordered list is created with the <ul> element

<li></li> Each item in the list is placed between an opening <li> tag and a closing </li> tag. (The li stands for list item.)

### DEFINITION LISTS

<dl></dl> The definition list is created with the <dl> element and usually consists of a series of terms and their definitions. Inside the <dl> element you will usually see pairs of <dt> and <dd> elements.

<dt></dt> This is used to contain the term being defined (the definition term).

<dd></dd> This is used to contain the definition.

### NESTED LISTS

You can put a second list inside an <li> element to create a sublist or nested list

## CHAPTER 4: LINKS

### LINKING TO OTHER SITES

<a href=""></a> Links are created using the <a> element which has an attribute called href. The value of the href attribute is the page that you want people to go to when they click on the link.

When you link to a different website, the value of the href attribute will be the full web

address for the site, which is known as an **absolute** URL.

### LINKING TO OTHER PAGES ON THE SAME SITES

When you are linking to other pages within the same site, you do not need to specify the domain name in the URL. You can use a shorthand known as a **relative** URL.

If all the pages of the site are in the same folder, then the value of the href attribute is just the name of the file.

### EMAIL LINKS

To create a link that starts up the user's email program and addresses an email to a specified email address, you use the <a> element. However, this time the value of the href attribute starts with mailto: and is followed by the email address you want the email to be sent to.

For example: <a href="mailto:pate.zero123@gmail.com"> My email</a>

### OPENING LINKS IN A NEW WINDOW

If you want a link to open in a new window, you can use the target attribute on the opening <a> tag. The value of this attribute should be \_blank.

For example:

<a href="http://www.imdb.com" target="\_blank">Internet Movie Database</a>

Generally you should avoid opening links in a new window.

### LINKING TO A SPECIFIC PART OF THE SAME PAGE

To link to an element that uses an id attribute you use the <a> element again, but the value of

the href attribute starts with the # symbol, followed by the value of the id attribute of the element you want to link to. For example, <a href="#top"> links to the <h1 id="top"></h1> element at the top of the page whose id attribute has a value of top

### LINKING TO A SPECIFIC PART OF ANOTHER PAGE

Therefore, the href attribute will contain the address for the page (either an **absolute URL** or

a **relative URL**), followed by the # symbol, followed by the value of the **id attribute** that is used on the element you are linking to.

For example: <a href="http:/www.htmlandcssbookcom/#bottom"></a>

# 

# 

## CHAPTER 5: IMAGES

### CHOOSING IMAGES FOR YOUR SITE

Image should:

* Be relevant
* Convey information
* Convey the right mood
* Be instantly recognisable
* Fit the color palette

Remember that all images are subject to copyright, and you can get in trouble for simply taking photographs from another website.

### STORING IMAGES ON YOUR SITE

As a website grows, keeping images in a separate folder helps you understand how the site is organized.

On a big site you might like to add subfolders inside the images folder. For example, images such as logos and buttons might sit in a folder called **interface**, product photographs might sit in a page called **products**, and images related to news might live in a folder called **news**.

### ADDING IMAGES

To add an image into the page you need to use an <img> element. This is an empty element (which means there is no closing tag). It must carry the following two attributes:

* src This tells the browser where it can find the image file. This will usually be a relative URL pointing to an image on your own site.
* alt This provides a text description of the image which describes the image if you cannot see it.
* title provides additional information about the image. Most browsers will display the content of this attribute in a tooltip when the user hovers over the image.

### HEIGHT & WIDTH OF IMAGES

You will also often see an <img> element use two other attributes that specify its size:

* height This specifies the height of the image in pixels
* widths This specifies the width of the image in pixels.

The size of images is increasingly being specified using CSS rather than HTML.

### WHERE TO PLACE IMAGES IN YOUR CODE

1. BEFORE A PARAGRAPH
2. INSIDE THE START OF PARAGRAPH
3. IN THE MIDDLE OF PARAGRAPH

### THREE RULES FOR CREATING IMAGES

1. SAVE IMAGES IN THE RIGHT FORMAT Websites mainly use images in jpeg, gif, or png format
2. SAVE IMAGES AT THE RIGHT SIZE If the image is larger than the width and height if you have specified, the image will take longer to display on the page.
3. USE THE CORRECT RESOLUTION Computer screens are made up of dots known as pixels. Images used on the web are also made up of tiny dots. Resolution refers to the number of dots per inch, and most computer screens only show web pages at 72 pixels per inch. So saving images at a higher resolution results in images that are larger than necessary and take longer to download.

### IMAGE RESOLUTION

Images created for the web should be saved at a resolution of 72 ppi. The higher the resolution of the image, the larger the size of the file.

### ANIMATED GIFS

Animated GIFs show several frames of an image in sequence and therefore can be used to create simple animations.

### EXAMINING IMAGES ON THE WEB

**CHROME**

Size: Open Image in New Tab

Size appears in new tab

Download: Save Image As

**FIREFOX**

Size: View Image Info

Size appears in pop-up window

Download: Save Image As

**INTERNET EXPLORER**

Size: Properties

Size appears in pop-up window

Download: Save Image

**SAFARI**

Size: Open Image in New Tab

Size appears in title bar

Download: Save Image As

### HTML5: FIGURE AND FIGURE CAPTION

<figure></figure> Images often come with captions. HTML5 has introduced a new <figure> element to contain images and their caption so that the two are associated. You can have more than one image inside the <figure> element as long as they all share the same caption.

<figcaption></figcaption> The <figcaption> element has been added to HTML5 in order to allow web page authors to add a caption to an image.

Older browsers that do not understand HTML5 elements simply ignore the new elements and display the content of them.

## CHAPTER 6: TABLES

### WHAT’S A TABLE

A table represents information in a grid format.

Grids allow us to understand complex data by referencing information on two axes.

Each block in the grid is referred to as a table cell. In HTML a table is written out row by row.

### BASIC TABLE STRUCTURE

<table></table> The <table> element is used to create a table. The contents of the table are written out row by row.

<tr></tr> You indicate the start of each row using the opening <tr> tag. (The tr stands for table row.) At the end of the row you use a closing </tr> tag.

<td></td> Each cell of a table is represented using a <td> element. (The td stands for table data.) At the end of each cell you use a closing </td> tag.

Some browsers automatically draw lines around the table and/or the individual cells.

### TABLE HEADINGS

<th></th> The <th> element is used just like the <td> element but its purpose is to represent the heading for either a column or a row. (The th stands for table heading.)

Browsers usually display the content of a <th> element in bold and in the middle of the cell.

### SPANNING COLUMNS

The colspan attribute can be used on a <th> or <td> element and indicates how many columns that cell should run across

### SPANNING ROWS

The rowspan attribute can be used on a <th> or <td> element to indicate how many rows a cell should span down the table

### LONG TABLES

There are three elements that help distinguish between the main content of the table and the first and last rows (which can contain different content)

* The headings of the table should sit inside the <thead> element
* The body should sit inside the <tbody> element
* The footer belongs inside the <tfoot> element

## CHAPTER 7: FORMS

### FORM CONTROLS

There are several types of form controls that you can use to collect information from visitors to your site:

* ADDING TEXT
  + Text input (single-line)
  + Password input
  + Text area (multi-line)
* MAKING CHOICES
  + Radio buttons
  + Checkboxes
  + Drop-down boxes
* SUBMITTING FORMS
  + Submit buttons
  + Image buttons
* UPLOADING FILES

### FORM STRUCTURE

<form></form> Form controls live inside a <form> element. This element should always carry the action attribute and will usually have a method and id attribute too.

action Every <form> element requires an action attribute. Its value is the URL for the page on the server that will receive the information in the form when it is submitted.

method Forms can be sent using one of two methods: get or post

* With the get method, the values from the form are added to the end of the URL specified in the action attribute. The get method is ideal for:
  + Short forms (such as search boxes)
  + When you retrieving data from web server (not sending information that should be added to or deleted from a database)
* With the post method the values are sent in what are known as HTTP headers. As a rule of thumb you should use the post method if your form:
  + Allows users to upload a file
  + Is very long
  + contains sensitive data (e.g. passwords)
  + adds information to, or deletes information from, a database
* If the method attribute is not used, the form data will be sent using the get method

id the value is used to identify the form distinctly from other elements on the page.

### TEXT INPUT

<input /> The <input> element is used to create several different form controls. The value of the type attribute determines what kind of input they will be creating.

type="text" When the type attribute has a value of text, it creates a single line text input.

name Each form control requires a name attribute. The value of this attribute identifies the form control and is sent along with the information they enter to the server.

maxlength Its value is the number of characters they may enter.

### PASSWORD INPUT

<input type="password" /> When the type attribute has a value of password it creates a text box that acts just like a single-line text input, except the characters are blocked out.

name and maxlength are the same as **Text input.**

### TEXT AREA

<textarea></textarea> The <textarea> element is used to create a mutli-line text input. Unlike other input elements this is not an empty element. It should therefore have an opening and a closing tag.

Any text that appears between the opening <textarea> and closing </textarea> tags will appear in the text box when the page loads.

If you are creating a new form, you should use CSS to control the width and height of a <textarea>.

The rows attribute indicates how many rows the text area should take up vertically.

### RADIO BUTTON

<input type="radio" /> Radio buttons allow users to pick just one of a number of options.

name The name attribute is sent to the server with the value of the option the user selects.

value The value attribute indicates the value that is sent to the server for the selected option. The value of each of the buttons in a group should be different.

checked The checked attribute can be used to indicate which value (if any) should be selected when the page loads. The value of this attribute is checked. Only one radio button in a group should use this attribute.

**Noted**: Once a radio button has been selected it cannot be deselected. The user can only select a different option. If you are only allowing the user one option and want them to be able to deselect it (for example if they are indicating they agree to terms and conditions), you should use a checkbox instead.

### CHECKBOX

<input type="checkbox" /> Checkboxes allow users to select (and unselect) one or more options in answer to a question.

name The name attribute is sent to the server with the value of the option(s) the user selects. The value of the name attribute should be the same for all of the buttons that answer that question

value The value attribute indicates the value sent to the server if this checkbox is checked.

checked The checked attribute indicates that this box should be checked when the page loads. If used, its value should be checked.

### DROP DOWN LIST BOX

<select></select> A drop down list box (also known as a select box) allows users to select one option from a drop down list. The <select> element is used to create a drop down list box. It contains two or more <option> elements.

name The name attribute indicates the name of the form control being sent to the server, along with the value the user selected.

<option></option> The <option> element is used to specify the options that the user can select from.

* value The <option> element uses the value attribute to indicate the value that is sent to the server along with the name of the control if this option is selected.
* selected The selected attribute can be used to indicate the option that should be selected when the page loads. The value of this attribute should be selected. If this attribute is not used, the first option will be shown when the page loads. If the user does not select an option, then the first item will be sent to the server as the value for this control.

The function of the drop down list box is similar to that of the radio buttons (in that only one option can be selected). There are two key factors in choosing which to use:

1. If users need to see all options at a glance, radio buttons are better suited.
2. . If there is a very long list of options (such as a list of countries), drop down list boxes work better.

### MULTIPLE SELECT BOX

<select size="" multiple="multiple"></select>

size Its value should be the number of options you want to show at once.

multiple You can allow users to select multiple options from this list by adding the multiple attribute with a value of multiple.

### FILE INPUT BOX

<input type="file" /> If you want to allow users to upload a file (for example an image, video, mp3, or a PDF), you will need to use a file input box.

type="file" This type of input creates a box that looks like a text input followed by a browse button. When the user clicks on the browse button, a window opens up that allows them to select a file from their computer to be uploaded to the website.

When you are allowing users to upload files, the method attribute on the <form> element must have a value of post. (You cannot send files using the HTTP get method.)

When a user clicks on the browse button, the presentation of the window that allows them to browse for the file they want to upload will match the windows of the user's operating system. You cannot control the appearance of these windows.

### SUBMIT BUTTON

<input type="submit" /> The submit button is used to send a form to the server.

name It can use a name attribute but it does not need to have one.

value The value attribute is used to control the text that appears on a button.

### IMAGE BUTTON

<input type="image" /> If you want to use an image for the submit button, you can give the type attribute a value of image. The src, width, height, and alt attributes work just like they do when used with the <img> element.

### BUTTON & HIDDEN CONTROLS

<button></button> The <button> element was introduced to allow users more control over how their buttons appear, and to allow other elements to appear inside the button. You can combine text and images between the opening <button> tag and closing </button> tag.

<input type="hidden" /> allow web page authors to add values to forms that users cannot see.

### LABELLING FORM CONTROLS

<label></label> each form control should have its own <label> element as this makes the form accessible to vision-impaired users. The <label> element can be used in two ways. It can:

1. Wrap around both the text description and the form input.
2. Be kept separate from the form control and use the for attribute to indicate which form control it is a label for.

for The value of the for attribute matches that of the id attribute on the form control it is labelling. This technique using the for and id attributes can be used on any form control. When a <label> element is used with a checkbox or radio button, users can click on either the form control or the label to select.

### GROUPING FORM ELEMENTS

<fieldset></fieldset> You can group related form controls together inside the <fieldset> element. This is particularly helpful for longer forms.

<legend></legend> The <legend> element can come directly after the opening <fieldset> tag and contains a caption which helps identify the purpose of that group of form controls.

### HTML5: FORM VALIDATION

HTML5 is introducing validation and leaving the work to the browser.

Validation helps ensure the user enters information in a form that the server will be able to understand when the form is submitted. Validating the contents of the form before it is sent to the server.

### HTML5: DATE INPUT

<input type="date" /> HTML5 introduces new form controls to standardize the way that some information is gathered. Older browsers that do not recognize these inputs will just treat them as a single line text box.

### HTML5: EMAIL & URL INPUT

<input type="email" /> Browsers that support HTML5 validation will check that the user has provided information in the correct format of an email address. Some smart phones also optimize their keyboard to display the keys you are most likely to need when entering an email address.

<input type="url" /> Browsers that support HTML5 validation will check that the user has provided information in the format of a URL. Some smart phones also optimize their keyboard to display the keys you are most likely to need when entering a URL.

### HTML5: SEARCH INPUT

<input type="search" /> Recent browsers add some features that improve usability. For example, Safari on a Mac adds a cross to clear the search box when you have started to enter information. Safari also automatically rounds the corners on the search input field.

placeholder On any text input, you can also use an attribute called placeholder whose value is text that will be shown in the text box until the user clicks in that area. Older browsers simply ignore this attribute.

## CHAPTER 8: EXTRA MARKUP

### DOCTYPES

Because there have been several versions of HTML, each web page should begin with a DOCTYPE declaration to tell a browser which version of HTML the page is using.

HTML 5: <!DOCTYPE html>

HTML 4: <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

### COMMENTS IN HTML

<!-- comment goes here --> comments are not visible to users in the main browser window.

### ID ATTRIBUTE

Every HTML element can carry the id attribute. It is used to uniquely identify that element from other elements on the page.

It is important that no two elements on the same page have the same value for their id attributes.

The id attribute is known as a **global attribute** because it can be used on any element.

### CLASS ATTRIBUTE

Every HTML element can also carry a class attribute. Sometimes, rather than uniquely identifying one element within a document, you will want a way to identify several elements as being different from the other elements on the page.

The class attribute on any element can share the same value.

### BLOCK ELEMENTS

Some elements will always appear to start on a new line in the browser window. These are known as **block level** elements.

Examples of block elements are <h1>, <p>, <ul>, and <li>.

### INLINE ELEMENTS

Some elements will always appear to continue on the same line as their neighbouring elements. These are known as **inline** elements.

Examples of inline elements are <a>, <b>, <em>, and <img>

### GROUPING TEXT & ELEMENTS IN A BLOCK

<div></div> The <div> element allows you to group a set of elements together in one block-level box. It will make no difference to the presentation of the page.

### GROUPING TEXT & ELEMENTS INLINE

<span></span> The <span> element acts like an inline equivalent of the <div> element. It is used to either:

1. Contain a section of text where there is no other suitable element to differentiate it from its surrounding text.
2. Contain a number of inline elements

The most common reason why people use <span> elements is so that they can control the appearance of the content of these elements using CSS.

### IFRAMES

<**iframe** src=""></**iframe**> An iframe is like a little window that has been cut into your page — and in that window you can see another page. . The content of the iframe can be any html page (either located on the same server or anywhere else on the web).

src The src attribute specifies the URL of the page to show in the frame.

height The height attribute specifies the height of the iframe in pixels.

width The width attribute specifies the width of the iframe in pixels.

scrolling The scrolling attribute will not be supported in HTML5. It indicates whether the iframe should have scrollbars or not.

frameborder The frameborder attribute will not be supported in HTML5. It indicates whether the frame should have a border or not.

seamless In HTML5, a new attribute called seamless can be applied to an iframe where scrollbars are not desired. The seamless attribute (like some other new HTML5 attributes) does not need a value, but you will often see authors give it a value of seamless. Older browsers do not support the seamless attribute.

### INFORMATION ABOUT YOUR PAGES

<meta /> The <meta> element lives inside the <head> element and contains information about that web page. It is not visible to users but fulfills a number of purposes such as telling search engines about your page, who created it, and whether or not it is time sensitive. (If the page is time sensitive, it can be set to expire.)

The value of the name attribute can be anything you want it to be. Some defined values for this attribute that are commonly used are:

* description This contains a description of the page. This description is commonly used by search engines to understand what the page is about and should be a maximum of 155 characters. Sometimes it is also displayed in search engine results.
* keyword This contains a list of comma separated words that a user might search on to find the page. In practice, this no longer has any noticeable effect on how search engines index your site.
* robots This indicates whether search engines should add this page to their search results or not. A value of noindex can be used if this page should not be added. A value of nofollow can be used if search engines should add this page in their results but not any pages that it links to.

The <meta> element also uses the http-equiv and content attributes in pairs:

* author This defines the author of the web page.
* pragma This prevents the browser from caching the page. (That is, storing it locally to save time downloading it on subsequent visits.)
* expires Because browsers often cache the content of a page, the expires option can be used to indicate when the page should expire (and no longer be cached). Note that the date must be specified in the format shown

### ESCAPE CHARACTERS

There are some characters that are used in and reserved by HTML code.

When using escape characters, it is important to check the page in your browser to ensure that the correct symbol shows up. This is because some fonts do not support all of these characters and you might therefore need to specify a different font for these characters in your CSS code.

**< Less-than sign &lt; &#60**

**> Greater-than sign &gt; &amp;**

**& Ampersand &amp; &#38;**

**“ Quotation mark &quot; &#34;**

**¢ Cent sign &cent; &#162;**

**£ Pound sign &pound; &#163;**

**¥ Yen sign &yen; &#165;**

**€ Euro sign &euro; &#8364;**

**© Copyright symbol &copy; &#169;**

**® Registered trademark &reg; &#174;**

**™ Trademark &trade; &#8482;**

**‘ Left single quote &lsquo; &#8216;**

**’ Right single quote &rsquo; &#8217;**

**“ Left double quotes &ldquo; &#8220;**

**” Right double quotes &rdquo; &#8221;**

**× Multiplication sign &times; &#215;**

**÷ Division sign &divide; &#247;**

## CHAPTER 9: FLASH, VIDEO & AUDIO

### HOW FLASH WORK

When you export the movie into SWF format, Flash creates code that you can use to embed the Flash movie in your page.

To view Flash, browsers need to use a plugin (an extra piece of software that runs in the browser) called the Flash Player.

### USE OF FLASH

In 2008, browsers started to support HTML5 <video> and <audio> tags. At the time of writing, Flash is still a popular way of playing video and audio on the web but more and more people are switching to HTML5

### HTML5: PREPARING VIDEO FOR YOUR PAGES

**SUPPORT** The new HTML5 <video> element is only supported by recent browsers.

**DIGITAL RIGHTS** the <video> element does not support any type of Digital Rights Management (DRM — sometimes referred to as copy protection)

**FORMATS** Not all browsers support the same video formats. Therefore, you need to supply your video in more than one format. To reach as many browsers as possible, you should provide the video in the following formats:

* **H264**: IE and Safari
* **WebM**: Android, Chrome, Firefox, Opera

**CONTROLS** The browser supplies its own controls for the player, and these can vary from browser to browser. You can control the appearance of these controls using JavaScript.

**IN THE BROWSER** One of the problems with players such as the Flash Player is that they can behave inconsistently when elements such as menus drop over them, or the window is scaled up or down. The HTML5 option solves these issues.

### HTML5: ADDING VIDEO TO YOUR PAGES

<**video**></**video**> The <video> element has a number of attributes which allow you to control video playback:

* src This attribute specifies the path to the video.
* poster This attribute allows you to specify an image to show while the video is downloading or until the user tells the video to play.
* width, height These attributes specify the size of the player in pixels
* controls When used, this attribute indicates that the browser should supply its own controls for playback.
* autoplay When used, this attribute specifies that the file should play automatically.
* loop When used, this attribute indicates that the video should start playing again once it has ended.
* preload This attribute tells the browser what to do when the page loads. It can have one of three values:
  + none The browser should not load the video until the user presses play.
  + auto The browser should download the video when the page loads.
  + metadata The browser should just collect information such as the size, first frame, track list, and duration.

In HTML5 you do not need to supply values for all attributes, such as the controls, autoplay, and loop attributes used with the <video> element.

If the browser does not support the <video> element or the format of video used, it will display whatever is between the opening <video> and closing </video> tags.

### HTML5: MULTIPLE VIDEO SOURCES

You can also use multiple <source /> elements to specify that the video is available in different formats.

### ADDING AUDIO TO WEB PAGES

By far the most popular format for putting audio on web pages is MP3.

HTML5 has introduced a new <audio> element. Browsers that support this element provide their own controls — much as they do for the video files we just looked at.

### HTML5: ADDING HTML5 AUDIO TO YOUR PAGES

<audio></audio> HTML5 introduced the <audio> element to include audio files in your pages. As with HTML5 video, browsers expect different formats for the audio. The <audio> element has a number of attributes which allow you to control audio playback:

* src This attribute specifies the path to the audio file.
* controls This attribute indicates whether the player should display controls. If you do not use this attribute, no controls will be shown by default. You can also specify your own controls using JavaScript.
* autoplay The presence of this attribute indicates that the audio should start playing automatically.
* preload This attribute indicates what the browser should do if the player is not set to autoplay.
* loop This attribute specifies that the audio track should play again once it has finished.

### HTML5: MULTIPLE AUDIO SOURCES

<source /> It is possible to specify more than one audio file using the <source> element between the opening <audio> and closing </audio> tags.

This is important because different browsers support different formats for audio files.

* **MP3**: Safari 5+, Chrome 6+, IE9
* **Ogg Vorbis**: Firefox 3.6, Chome6, Opera 1.5, IE9

CSS Note

# CHAPTER 10: Introducing CSS

## What is CSS?

Like HTML, CSS is not a programming language. It's not a markup language either. CSS is a style sheet language. CSS is what you use to selectively style HTML elements.

## Box Model

Everything displayed by CSS is a box

### Content and Sizing

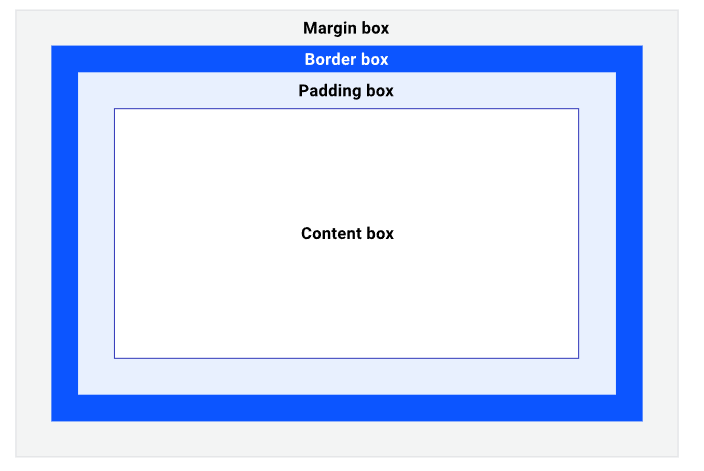
Boxes have different behavior based on their display value, their set dimensions, and the content that lives within them. This content could be even more boxes—generated by child elements—or plain text content. Either way, this content will affect the size of the box by default.

You can control this by using **extrinsic sizing,** or, you can continue to let the browser make decisions for you based on the content size, using **intrinsic sizing**.

**Key Term**

When content is too big for the box it is in, we call this **overflow**. You can manage how an element handles overflow content, using the overflow property.

### The areas of the box model



The four main areas of the box model: content box, padding box, border box and margin box.

* The **content box**, which is the area that the content lives in. This content can control the size of its parent, so is usually the most variably sized area.
* The **padding box** surrounds the content box and is the space created by the padding property. Because padding is inside the box, the background of the box will be visible in the space that it creates. If our box has overflow rules set, such as overflow: auto or overflow: scroll, the scrollbars will occupy this space too.
* The **border box** surrounds the padding box and its space is occupied by the border value. The border box is the bounds of your box and the **border edge** is the limit of what you can visually see. The border property is used to visually frame an element.
* The **margin box is** the space around your box, defined by the margin rule on your box. Properties such as outline and box-shadow occupy this space too because they are painted on top, so they don't affect the size of our box. You could have an outline-width of 200px on our box and everything inside and including the border box would be exactly the same size.

### Debugging the box model

Browser DevTools provides a visualisation of a selected box's box model calculations, which can help you understand how the box model works and importantly, how it is affecting the website you're working on.

## Selectors

To apply CSS to an element you need to select it. CSS provides you with a number of different ways to do this, and you can explore them in this module.



### Simple selectors

#### Universal selectors

A **universal selector**—also known as a wildcard—matches any element.

|  |
| --- |
| \* {  color: hotpink; } |

This rule causes every HTML element on the page to have hot pink text.

**Type selectors**

A **type selector** matches a HTML element directly.

|  |
| --- |
| section {  padding: 2em; } |

This rule causes every <section> element to have 2em of padding on all sides.

**Class selector**

An HTML element can have one or more items defined in their class attribute. The **class selector** matches any element that has that class applied to it.

|  |
| --- |
| <div class="my-class"></**div**> <button class="my-class"></**button**> <p class="my-class"></**p**> |

Any element that has the class applied to it will get colored red:

|  |
| --- |
| .my-class {  color: red; } |

Notice how the . is only present in CSS and **not** the HTML

**ID selector**

An HTML element with an id attribute should be the only element on a page with that ID value.

|  |
| --- |
| <div id="rad"></div> |

|  |
| --- |
| #rad {  border: 1px solid blue; } |

# CHAPTER 11: COLOR

### Numeric colors

#### **Hex colors:**

|  |
| --- |
| **h1 {  color: #b71540; }** |

Hexadecimal notation (often shortened to hex) is a shorthand syntax for RGB, which assigns a numeric value to red, green and blue, which are the three **primary colors**.

The hexadecimal ranges are **0-9** and **A-F.** When used in a six digit sequence, they are translated to the RGB numerical ranges which are 0-255 which correspond to the red, green, and blue color channels respectively.

You can also define an alpha value with any numerical colors. An alpha value is a percentage of transparency. In hex code, you add another two digits to the six digit sequence, making an eight digit sequence. For example, to set black in hex code, write #000000. To add a 50% transparency, change it to #00000080

* 0% alpha—which is fully transparent—is **00**: #00000000
* 50% alpha is **80**: #00000080
* 75% alpha is **BF**: #000000BF

#### **RGB (Red, Green, Blue)**

|  |
| --- |
| **h1 {  color: rgb(183, 21, 64); }** |

RGB colors are defined with the rgb() color function, using either numbers or percentages as parameters. The numbers need to be within the **0-255** range and the percentages are between **0% and 100%‌**. RGB works on the 0-255 scale, so 255 would be equivalent to 100%, and 0 to 0%.

To set black in RGB, define it as rgb(0 0 0), which is zero red, zero green and zero blue. Black can also be defined as rgb(0%, 0%, 0%). White is the exact opposite: rgb(255, 255, 255) or rgb(100%, 100%, 100%).

An alpha is set in rgb() in one of two ways. Either add a / **after** the red, green and blue parameters, or use the rgba() function. The alpha can be defined with a percentage or a decimal between 0 and 1. For example, to set a 50% alpha black in modern browsers, write: rgb(0 0 0 / 50%) or rgb(0 0 0 / 0.5). For wider support, using the rgba() function, write: rgba(0, 0, 0, 50%) or rgba(0, 0, 0, 0.5).

#### **HSL (Hue, Saturation, Lightness)**

|  |
| --- |
| h1 {  color: hsl(344, 79%, 40%); } |

HSL stands for hue, saturation and lightness. Hue describes the value on the color wheel, from 0 to 360 degrees, starting with red (being both 0 and 360). A hue of 180, or 50% would be in the blue range.

Saturation is how vibrant the selected hue is. A fully desaturated color (with a saturation of 0%) will appear grayscale. And finally, lightness is the parameter which describes the scale from white to black of added light. A lightness of 100% will always give you white.

Using the hsl() color function, you define a true black by writing hsl(0 0% 0%), or even hsl(0deg 0% 0%). This is because the hue parameter defines the degree on the color wheel, which if you use the number type, is **0-360**. You can also use the angle type, which is (**0deg**) or (**0turn**). Both saturation and lightness are defined with percentages.

### Color Keywords

There are [148 named colors in CSS](https://developer.mozilla.org/docs/Web/CSS/color_value#color_keywords). These are plain English names such as purple, tomato and goldenrod. Some of the most popular names, according to the [Web Almanac](https://almanac.httparchive.org/en/2019/css), are black, white, red, blue and gray.

### Aside from standard colors, there are also special keywords available:

* transparent is a fully transparent color. It is also the initial value of background-color.
* currentColor is the contextual computed dynamic value of the color property. If you have a text color of red and then set the border-color to be currentColor, it will also be red. If the element that you define currentColoron doesn't have a value for color defined, currentColor will be computed by the cascade instead

# Chapter 12: Text

## Typeface Terminology

* Serif -  Serif fonts have extra details on the ends of the main strokes of the letters. These details are known as serifs.
* Sans-serif – Sans-serif fonts have straight ends to letters, and therefore have a much cleaner design.
* Monospace - Every letter in a monospace (or fixed-width) font is the same width. (Non-monospace fonts have different widths.)

## Choosing a typeface for website

* When choosing a typeface, it is important to understand that a browser will usually only display it if it's installed on that user's computer.

## Techniques that offer a wider choice of typefaces

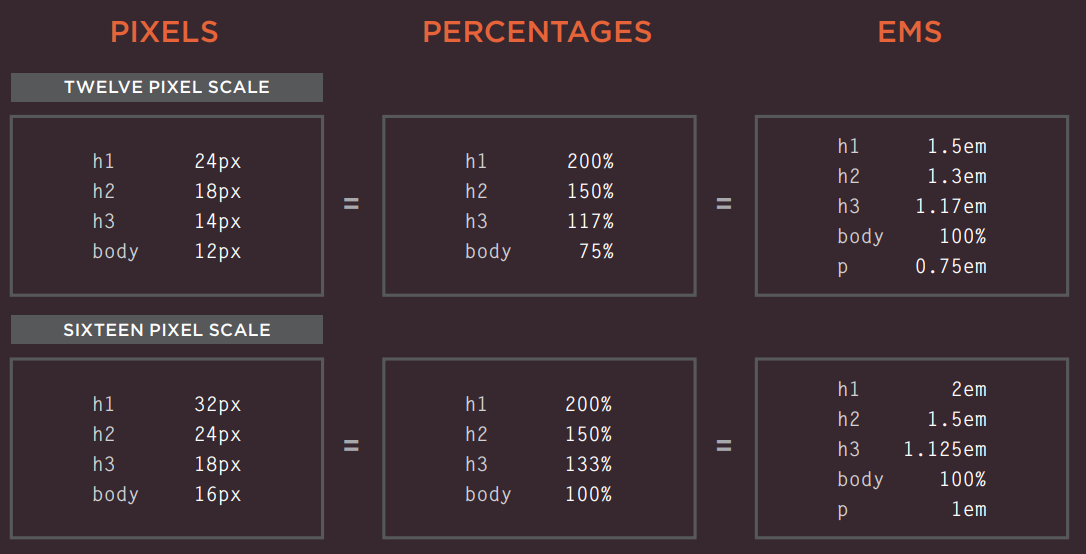
* The license to use the font must permit its distribution using @font-face.
* The service takes care of the licensing issues with the people who made the font.
* Note: If you design on a Mac, it is important to check what the typefaces look like on a PC because PCs can render type less smoothly. But if you design on a PC, then it should look fine on a Mac.

## Size of type

* Pixels - Pixels are commonly used because they allow web designers very precise control over how much space their text takes up. The number of pixels is followed by the letters px.
* Percentages - The default size of text in browsers is 16px. So a size of 75% would be the equivalent of 12px, and 200% would be 32px.
* EMS - An em is equivalent to the width of a letter m.



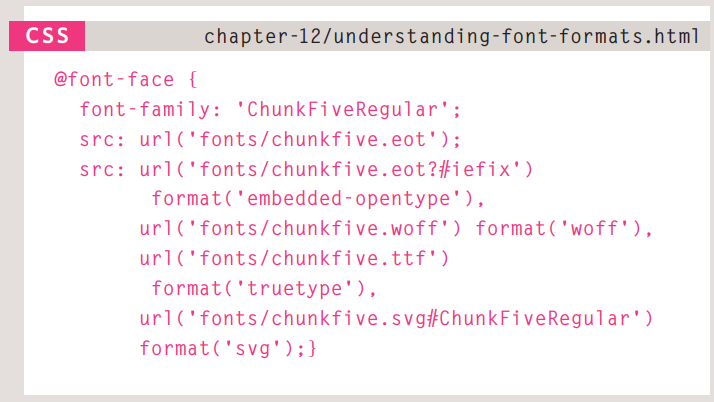
## Units of type size



* Ems allow you to change the size of text relative to the size of the text in the parent element. Since the default size of text in web browsers is 16 pixels, you can use similar rules to those shown for percentages.

## Understanding font formats

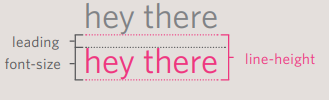
* Different browsers support different formats for fonts (in the same way that they support different audio and video formats), so you will need to supply the font in several variations to reach all browsers.
* The various font formats should appear in your code in this order:
* eot
* woff
* ttf/otf
* svg





## Line-height

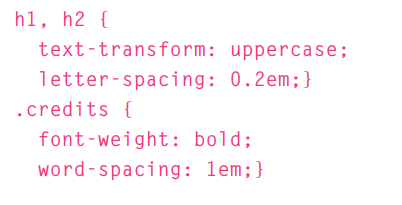
* Leading (pronounced ledding) is a term typographers use for the vertical space between lines of text. In a typeface, the part of a letter that drops beneath the baseline is called a descender, while the highest point of a letter is called the ascender. Leading is measured from the bottom of the descender on one line to the top of the ascender on the next



* In CSS, the line-height property sets the height of an entire line of text, so the difference between the fontsize and the line-height is equivalent to the leading (as shown in the diagram above).
* Increasing the line-height makes the vertical gap between lines of text larger.

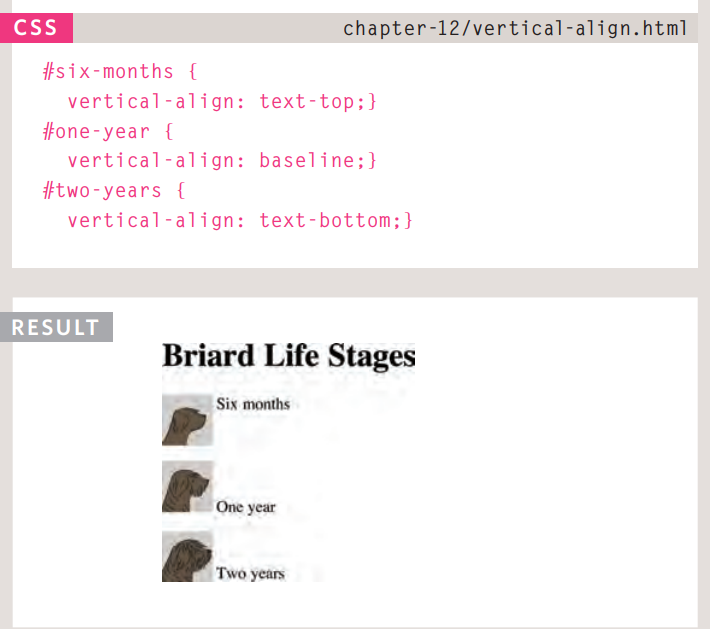
## Letter – word spacing

* You can control the space between each letter with the letter-spacing property
* You can also control the gap between words using the word-spacing property.
* The default gap between words is set by the typeface (often around 0.25em), and it is unlikely that you would need to change this property regularly



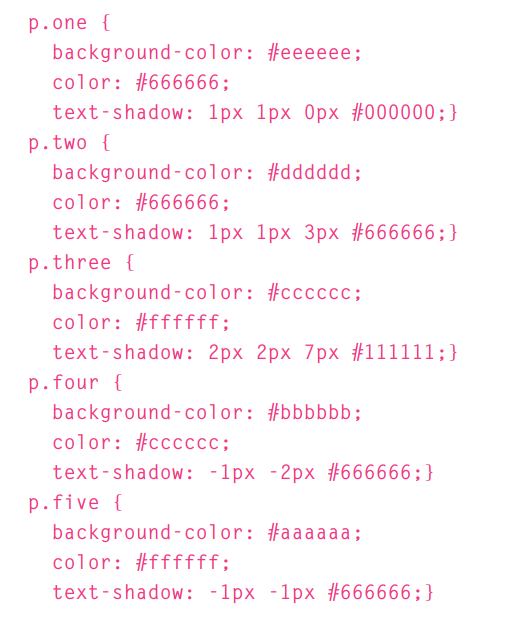
## Vertical-align

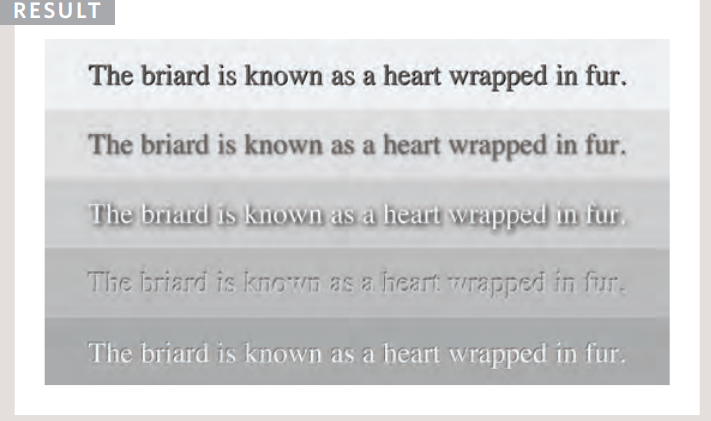
* It is not intended to allow you to vertically align text in the middle of block level elements such as <p> and <div>, although it does have this effect when used with table cells (the <td> and <th> elements).
* The values it can take are:
* baseline
* sub
* super
* top
* text-top
* middle
* bottom
* text-bottom



## Text-shadow

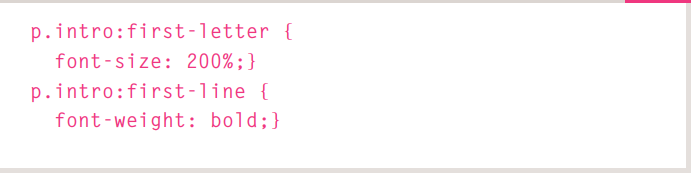
* The text-shadow property has become commonly used despite lacking support in all browsers.
* The value of this property is quite complicated because it can take three lengths and a color for the drop shadow.
* The first length indicates how far to the left or right the shadow should fall.
* The second value indicates the distance to the top or bottom that the shadow should fall.
* The third value is optional and specifies the amount of blur that should be applied to the drop shadow.
* The fourth value is the color of the drop shadow.
* The text-shadow property has become very popular but at the time of writing it was not supported in any versions of Internet Explorer (currently IE9). Other browser makers introduced it in Firefox 3.1, Safari 3, Chrome 2 and Opera 9.5.

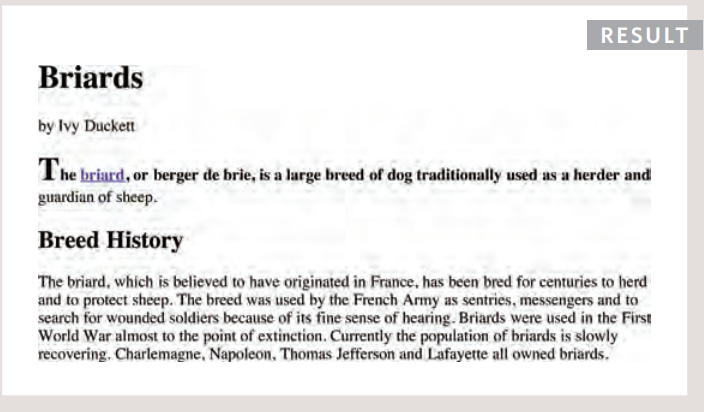




## First letter or line

* You can specify different values for the first letter or first line of text inside an element using :first-letter and :first-line.
* Technically these are not properties. They are known as pseudo-elements.





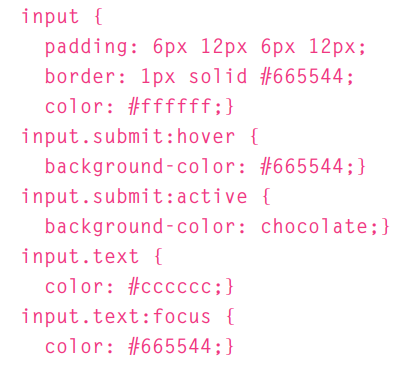
## Styling links

* In CSS, there are two pseudoclasses that allow you to set different styles for links that have and have not yet been visited.
* :link - This allows you to set styles for links that have not yet been visited.
* :visited - This allows you to set styles for links that have been clicked on.



## Responding to users

* :hover - This is applied when a user hovers over an element with a pointing device such as a mouse
* :active - This is applied when an element is being activated by a user
* :focus - This is applied when an element has focus



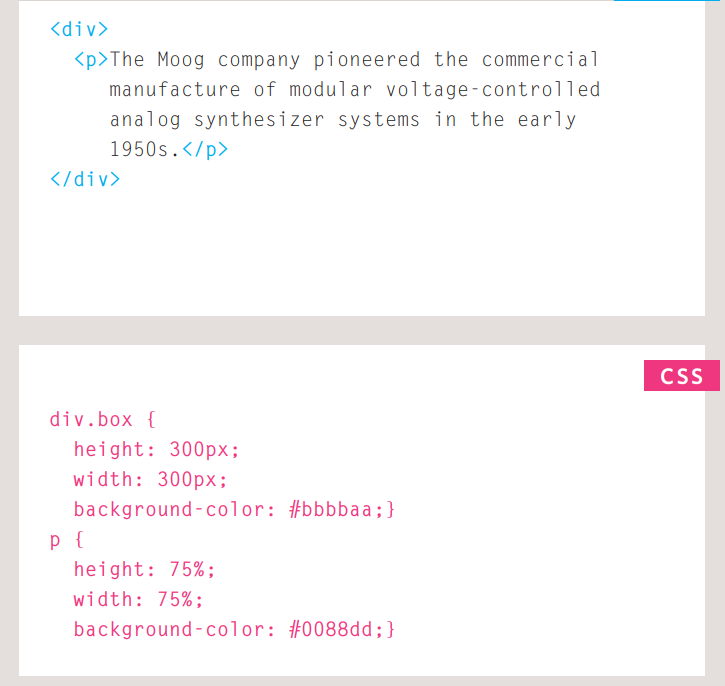


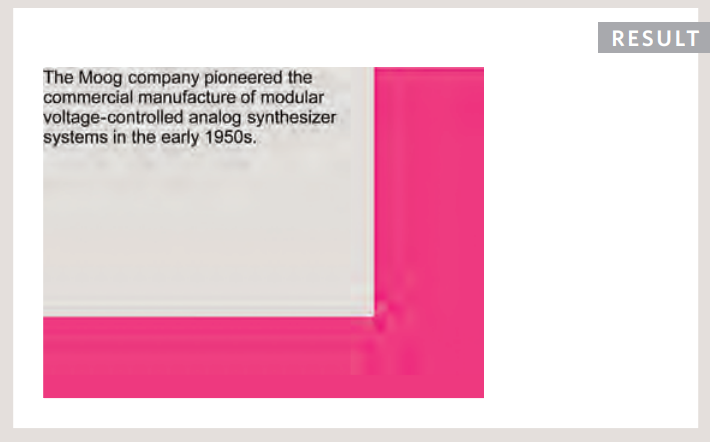
* There are also a set of attribute selectors that allow you to create rules that apply to elements that have an attribute with a specific value.

# Chapter 13: Boxes

## Box Dimensions

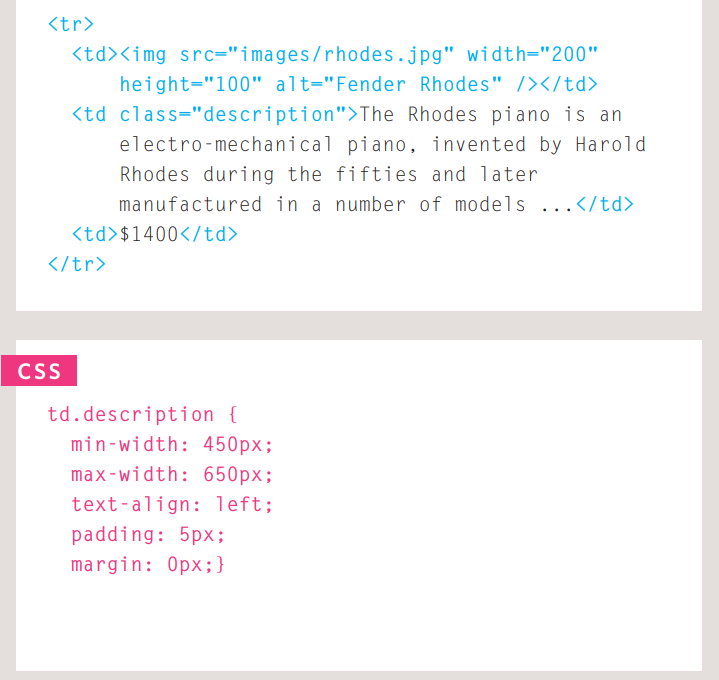
* The most popular ways to specify the size of a box are to use pixels, percentages, or ems
* When you use percentages, the size of the box is relative to the size of the browser window or, if the box is encased within another box, it is a percentage of the size of the containing box. When you use ems, the size of the box is based on the size of text within it. Designers have recently started to use percentages and ems more for measurements as they try to create designs that are flexible across devices which have different-sized screens.

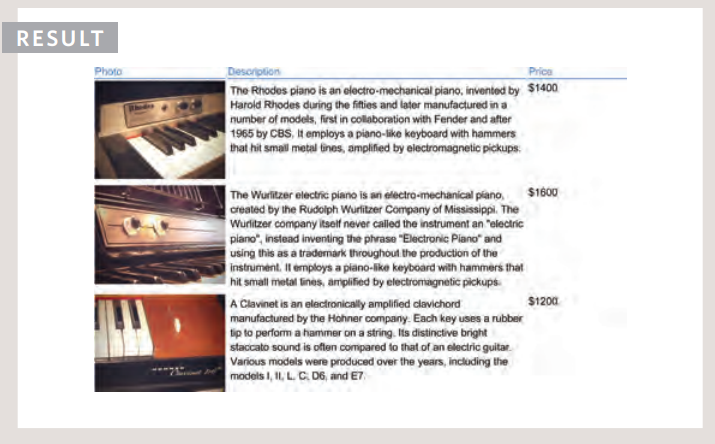




## Limiting width

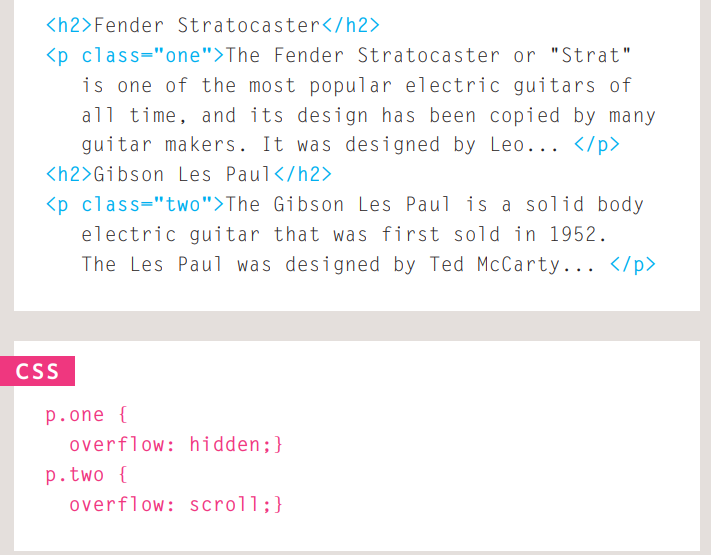
* Some page designs expand and shrink to fit the size of the user's screen. In such designs, the min-width property specifies the smallest size a box can be displayed at when the browser window is narrow, and the max-width property indicates the maximum width a box can stretch to when the browser window is wide.
* These are very helpful properties to ensure that the content of pages are legible (especially on the smaller screens of handheld devices).

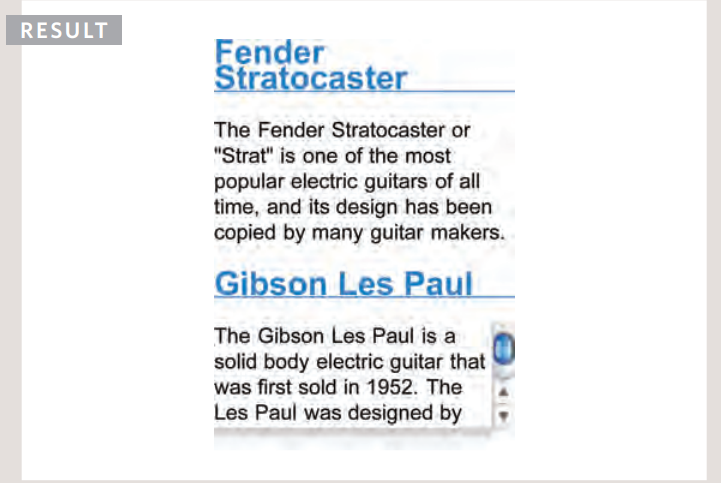




## Overflowing content

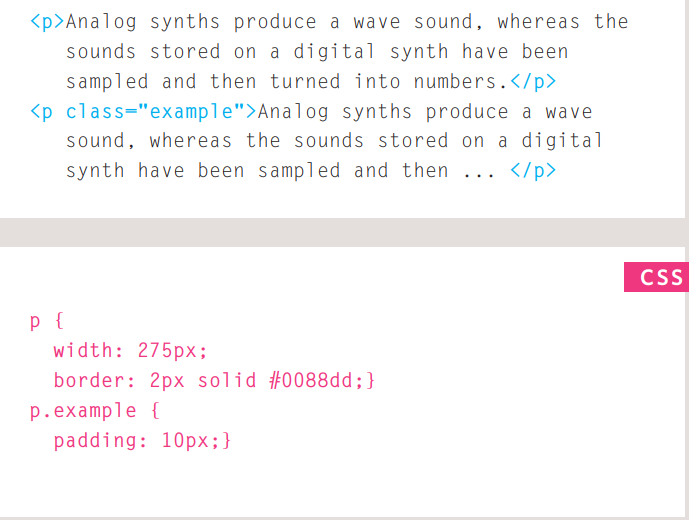
* It can have one of two values:
* hidden - This property simply hides any extra content that does not fit in the box.
* scroll - This property adds a scrollbar to the box so that users can scroll to see the missing content.
* The overflow property is particularly handy because some browsers allow users to adjust the size of the text to appear as large or as small as they want

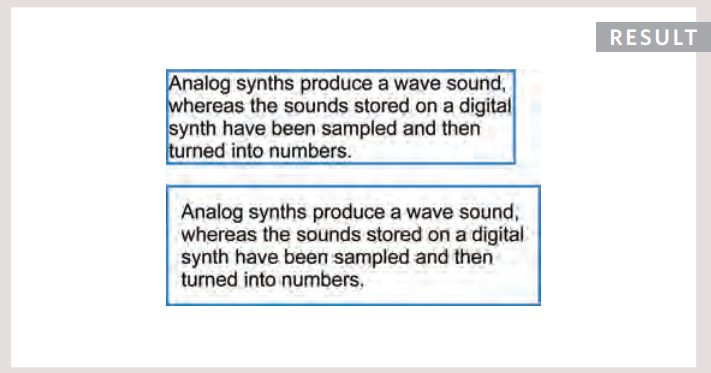




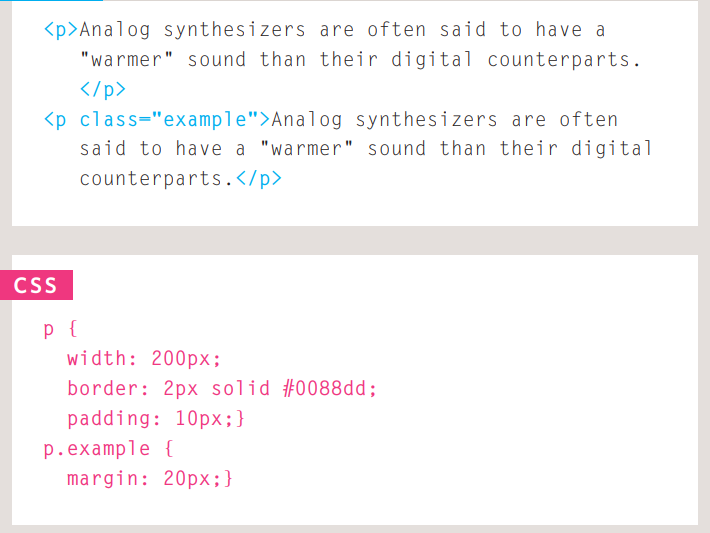
## Border, margin, padding

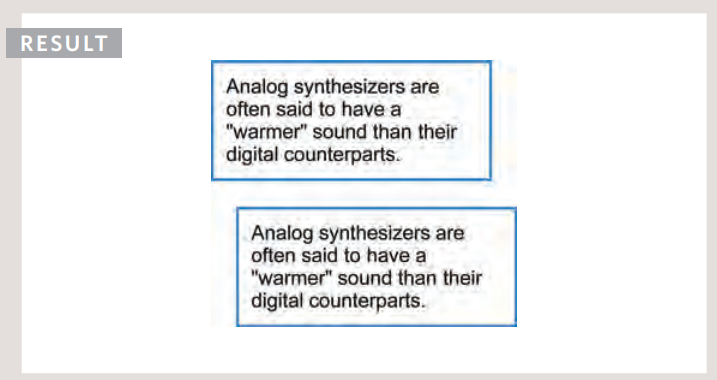
* Every box has three available properties that can be adjusted to control its appearance:
* Border - Every box has a border (even if it is not visible or is specified to be 0 pixels wide). The border separates the edge of one box from another.
* Margin – Margins sit outside the edge of the border. You can set the width of a margin to create a gap between the borders of two adjacent boxes.
* Padding - Padding is the space between the border of a box and any content contained within it.
* Note: If a width is specified for a box, padding is added onto the width of the box





* Note: If the width of a box is specified then the margin is added to the width of the box
* You can specify values for each side of a box using:
* margin-top
* margin-right
* margin-bottom
* margin-left





## Centering Content

* The text-align property is inherited by child elements. You therefore also need to specify the text-align property on the centered box if you do not want the text inside it to be centered.

## Hiding Boxes

* The visibility property allows you to hide boxes from users but It leaves a space where the element would have been.
* This property can take two values:
* hidden - This hides the element.
* visible - This shows the element.

## Box Shadows

* It must use at least the first of these two values as well as a color:
* Horizontal offset - Negative values position the shadow to the left of the box.
* Vertical Offset - Negative values position the shadow to the top of the box
* Blur Distance - If omitted, the shadow is a solid line like a border.
* Spread of shadow - If used, a positive value will cause the shadow to expand in all directions, and a negative value will make it contract.

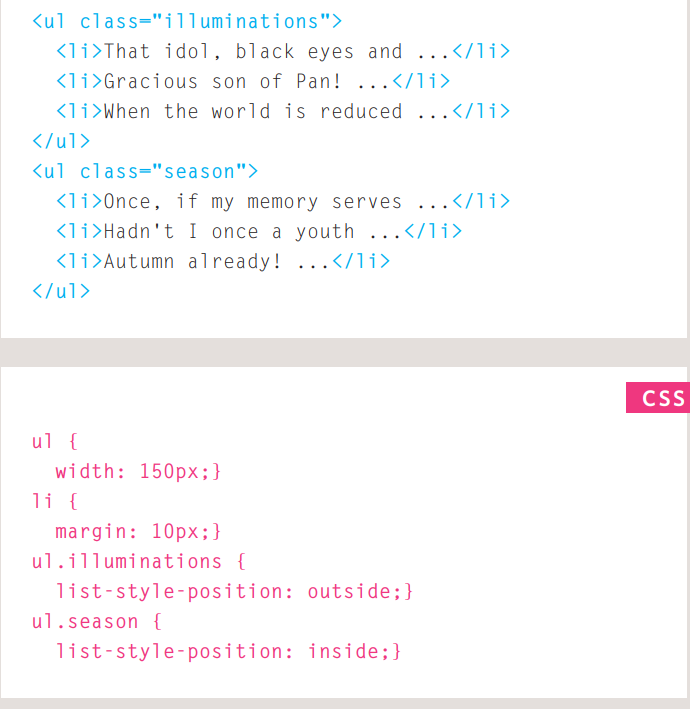
## Rounded corners

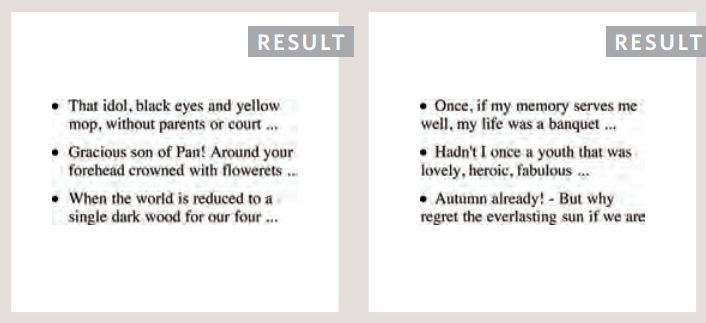
* Older browsers that do not support this property will show a box with right-angled corners.

# Chapter 14: Lists, Tables, and Forms

## Positioning the marker

* Lists are indented into the page by default and the list-styleposition property indicates whether the marker should appear on the inside or the outside of the box containing the main points.
* Outside - The marker sits to the left of the block of text. (This is the default behaviour if this property is not used.)
* Inside - The marker sits inside the box of text (which is indented).



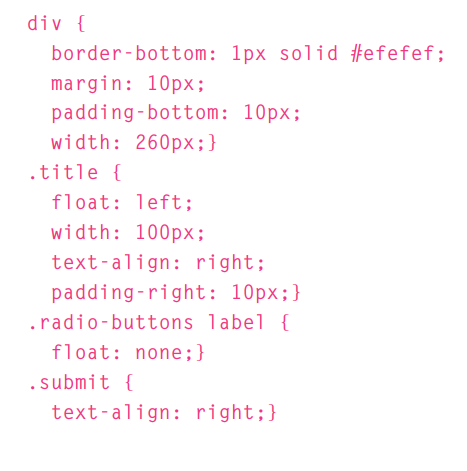


## Styling Forms

* when you come to look at a form in a few different browsers (as shown on the right), you will see that each browser displays them differently
* It is most common to style:
* Text inputs and text areas
* Submit buttons
* Labels on forms, to get the form controls to align nicely

## Aligning form controls: Solution

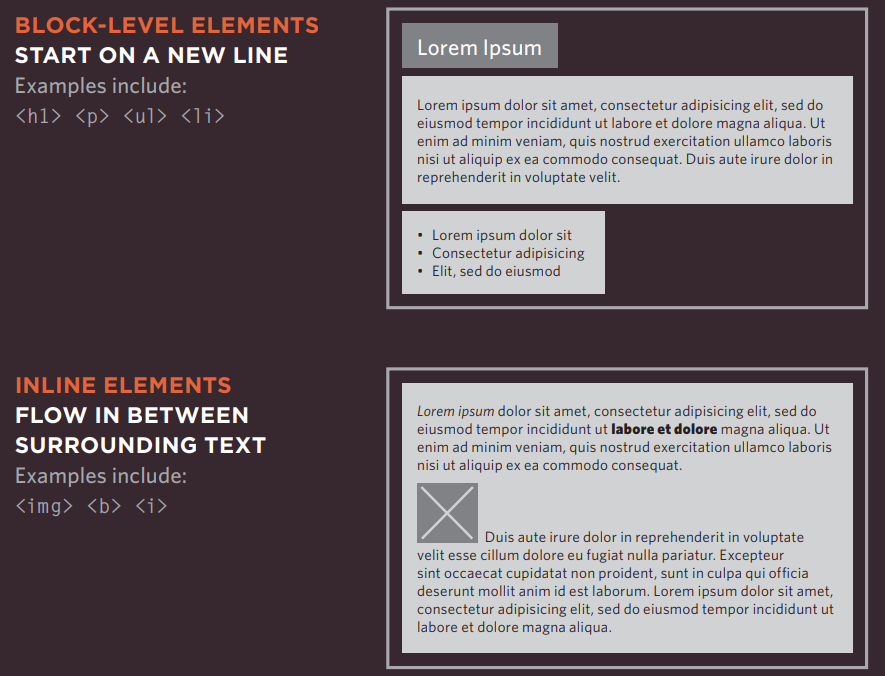
* Each row of the form has a title telling users what they need to enter.



# Chapter 15: Layout

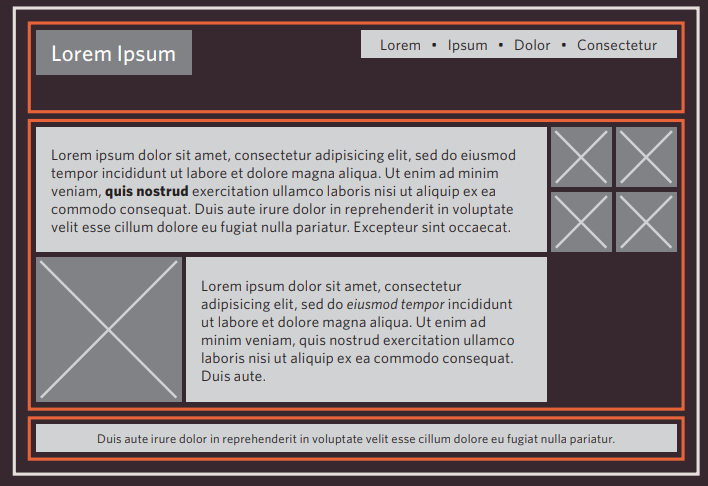
## Building blocks

* CSS treats each HTML element as if it is in its own box. This box will either be a block-level box or an inline box.
* Block-level boxes start on a new line and act as the main building blocks of any layout, while inline boxes flow between surrounding text.



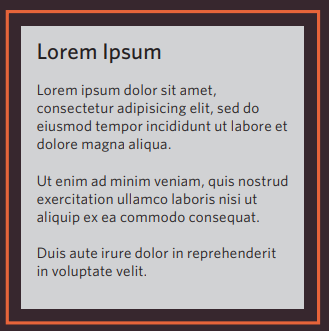
## Containing elements

* If one block-level element sits inside another block-level element then the outer box is known as the containing or parent element.
* A box may be nested inside several other block-level elements. The containing element is always the direct parent of that element.

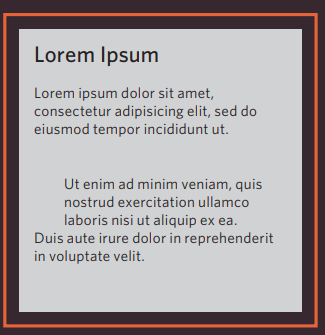


## Controlling the position of elements

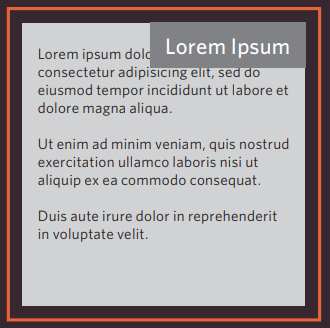
* Normal Flow -  Every block-level element appears on a new line, causing each item to appear lower down the page than the previous one. Even if you specify the width of the boxes and there is space for two elements to sit side-byside, they will not appear next to each other. This is the default behavior (unless you tell the browser to do something else)



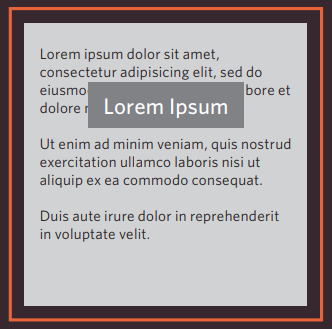
* The paragraphs appear one after the other, vertically down the page.
* Relative Positioning - This moves an element from the position it would be in normal flow, shifting it to the top, right, bottom, or left of where it would have been placed. This does not affect the position of surrounding elements; they stay in the position they would be in in normal flow.



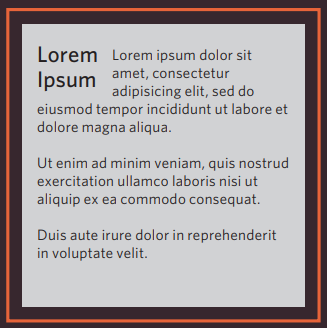
* The second paragraph has been pushed down and right from where it would otherwise have been in normal flow
* Absolute positioning - This positions the element in relation to its containing element. It is taken out of normal flow, meaning that it does not affect the position of any surrounding elements (as they simply ignore the space it would have taken up). Absolutely positioned elements move as users scroll up and down the page.



* The heading is positioned to the top right, and the paragraphs start at the top of the screen (as if the heading were not there).
* Fixed Positioning - This is a form of absolute positioning that positions the element in relation to the browser window, as opposed to the containing element. Elements with fixed positioning do not affect the position of surrounding elements and they do not move when the user scrolls up or down the page.



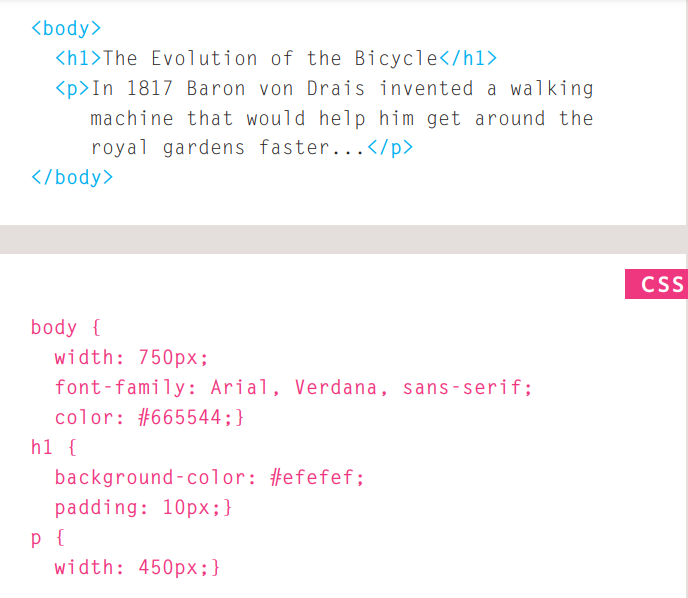
* The heading has been placed in the center of the page and 25% from the top of the screen. (The rest appears in normal flow.)
* Floating Elements - Floating an element allows you to take that element out of normal flow and position it to the far left or right of a containing box. The floated element becomes a block-level element around which other content can flow.

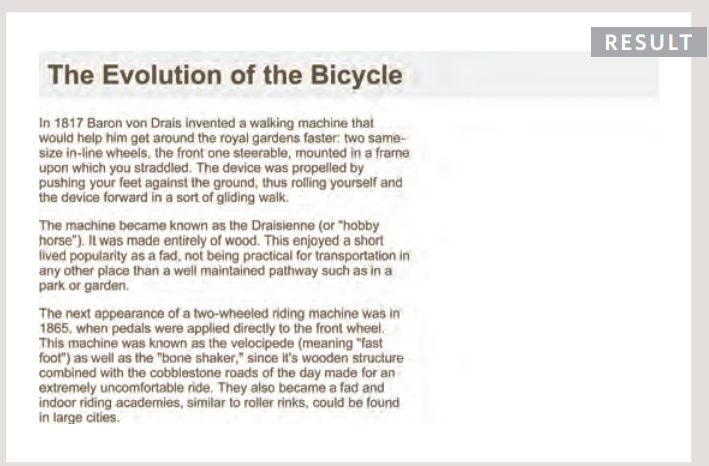


* The heading has been floated to the left, allowing the paragraphs of text to flow around it.
* The z-index property allows you to control which box appears on top.

## Normal Flow

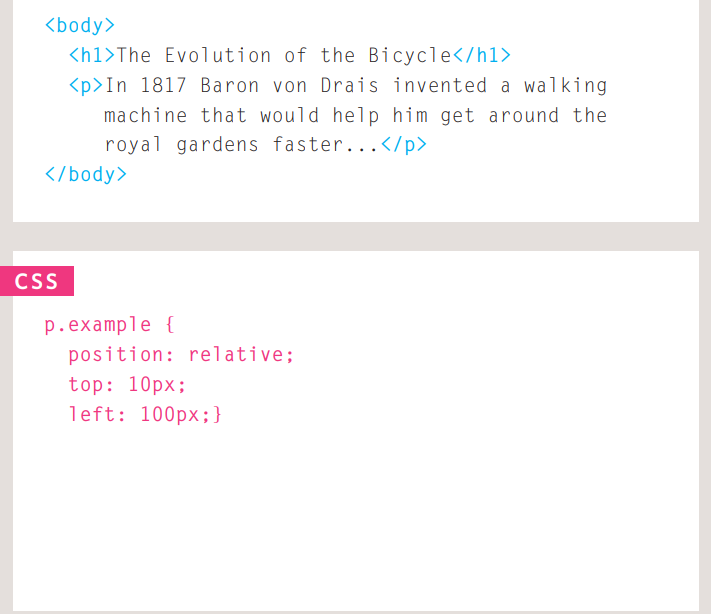
* In normal flow, each block-level element sits on top of the next one. Since this is the default way in which browsers treat HTML elements, you do not need a CSS property to indicate that elements should appear in normal flow, but the syntax would be:
* position: static:

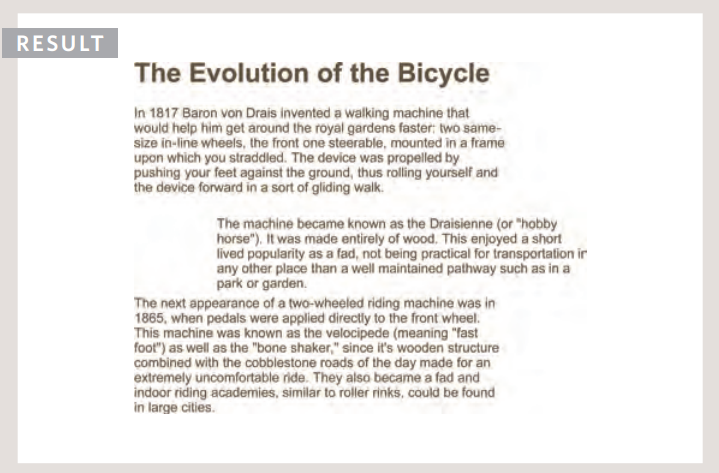




## Relative positioning

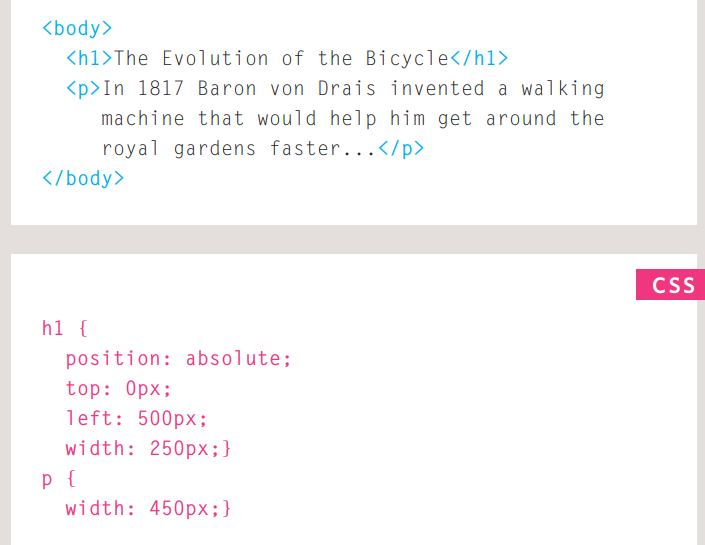
* position : relative
* Relative positioning moves an element in relation to where it would have been in normal flow.

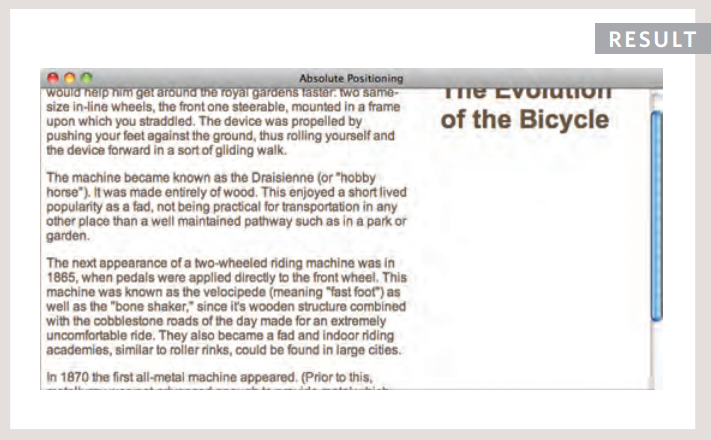




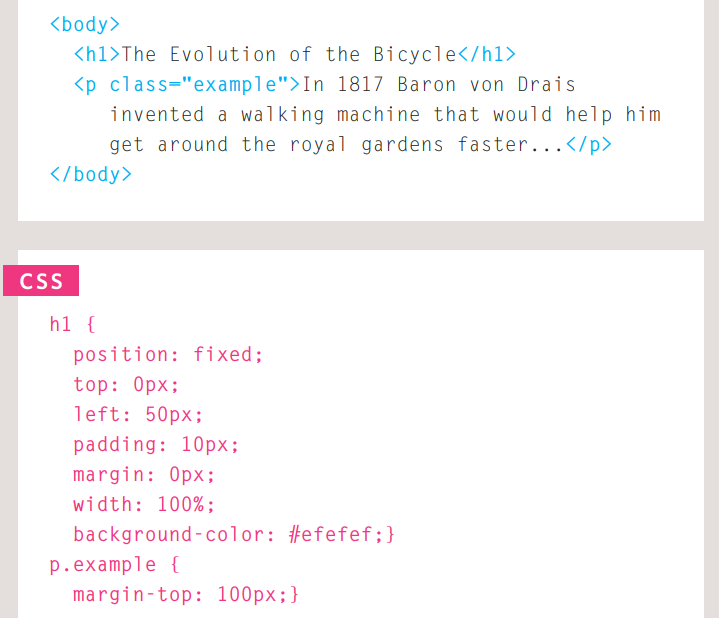
## Absolute Positioning

* position:absolute
* When the position property is given a value of absolute, the box is taken out of normal flow and no longer affects the position of other elements on the page. (They act like it is not there.)
* The box offset properties (top or bottom and left or right) specify where the element should appear in relation to its containing element





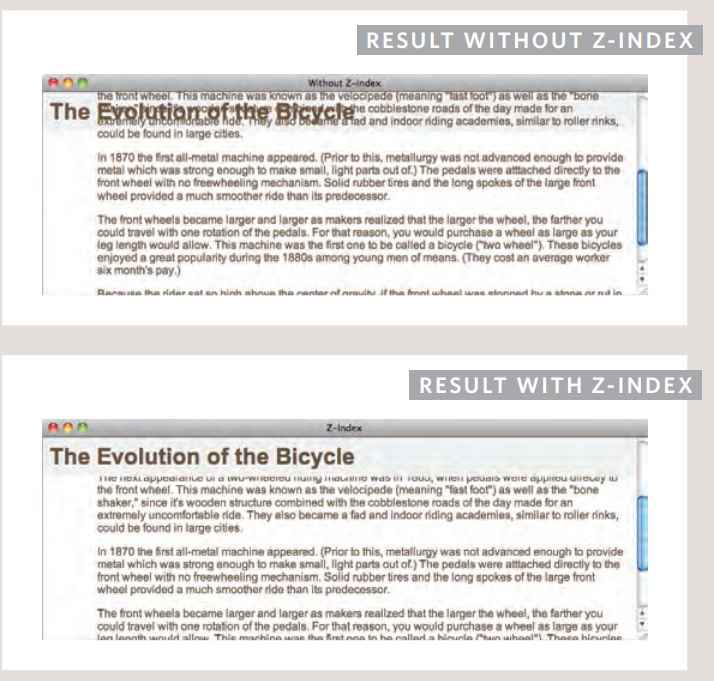
## Fixed Positioning

* Position:fixed
* Fixed positioning is a type of absolute positioning that requires the position property to have a value of fixed.
* It positions the element in relation to the browser window. Therefore, when a user scrolls down the page, it stays in the exact same place.
* 

## Overlapping elements

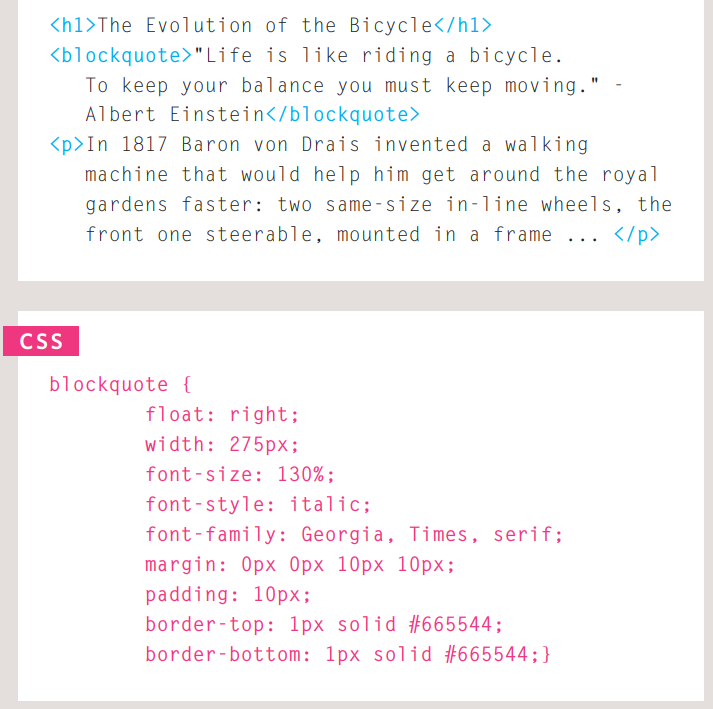
* z-index
* When you use relative, fixed, or absolute positioning, boxes can overlap. If boxes do overlap, the elements that appear later in the HTML code sit on top of those that are earlier in the page.
* If you want to control which element sits on top, you can use the z-index property

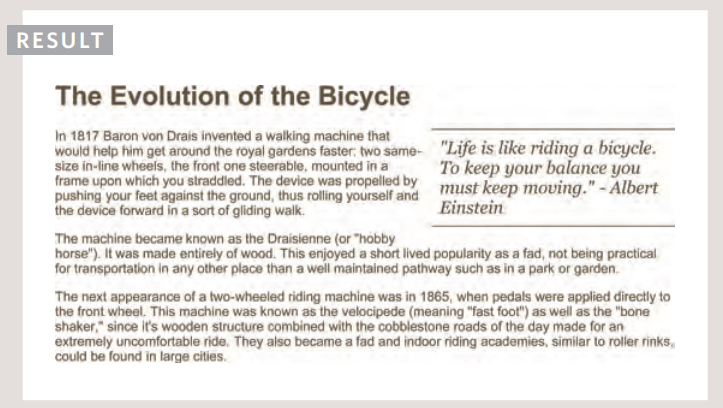




## Floating elements

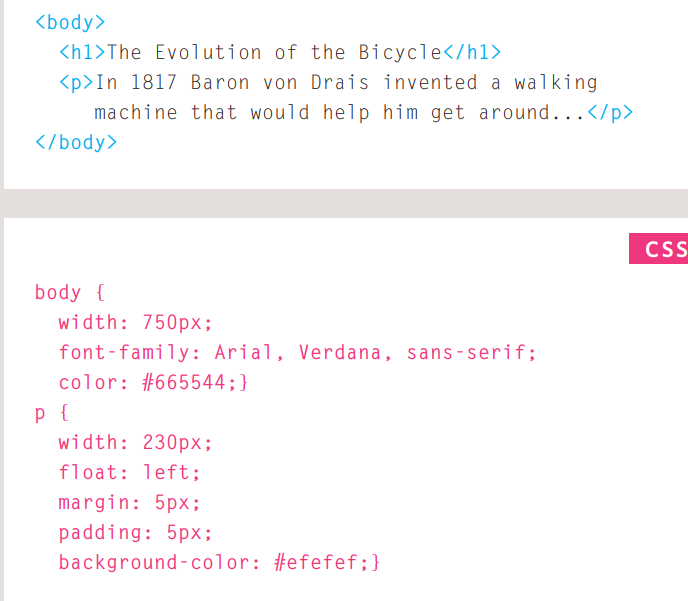
* When you use the float property, you should also use the width property to indicate how wide the floated element should be. If you do not, results can be inconsistent but the box is likely to take up the full width of the containing element (just like it would in normal flow).

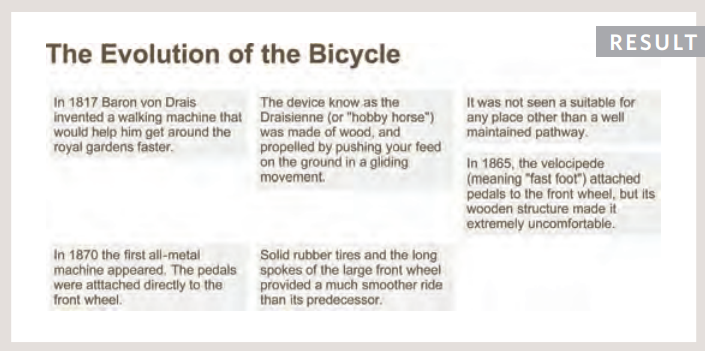




## Using float to place elements side-by-side

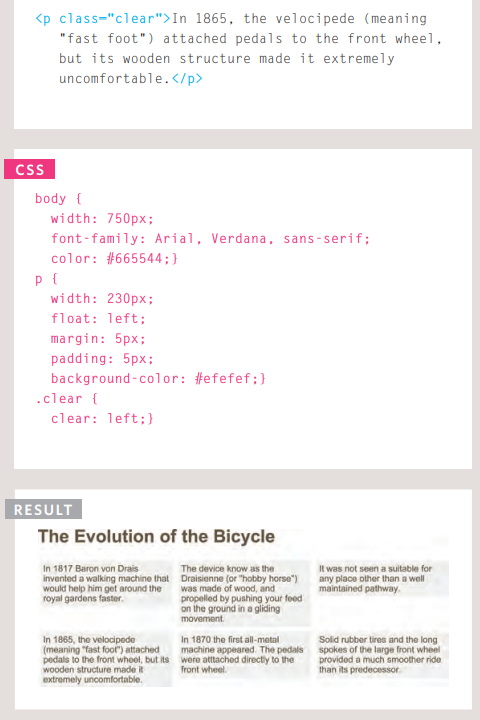
* A lot of layouts place boxes next to each other. The float property is commonly used to achieve this.





## Clearing floats

* The clear property allows you to say that no element (within the same containing element) should touch the left or righthand sides of a box. It can take the following values:
* left - The left-hand side of the box should not touch any other elements appearing in the same containing element.
* right - The right-hand side of the box will not touch elements appearing in the same containing element.
* both - Neither the left nor right-hand sides of the box will touch elements appearing in the same containing element.
* none - Elements can touch either side.



## Screen Sizes

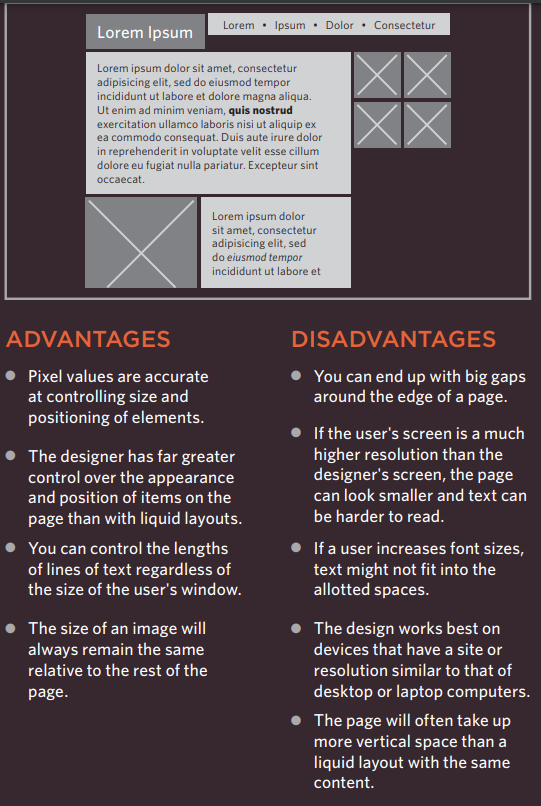
* Different visitors to your site will have different sized screens that show different amounts of information, so your design needs to be able to work on a range of different sized screens
* When designing for print, you always know the size of the piece of paper that your design will be printed on.

## Screen Resolution

* Resolution refers to the number of dots a screen shows per inch. Some devices have a higher resolution than desktop computers and most operating systems allow users to adjust the resolution of their screens.

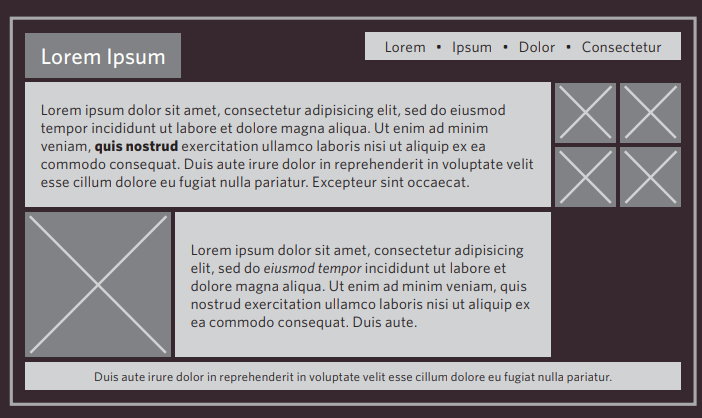
## Fixed width layouts

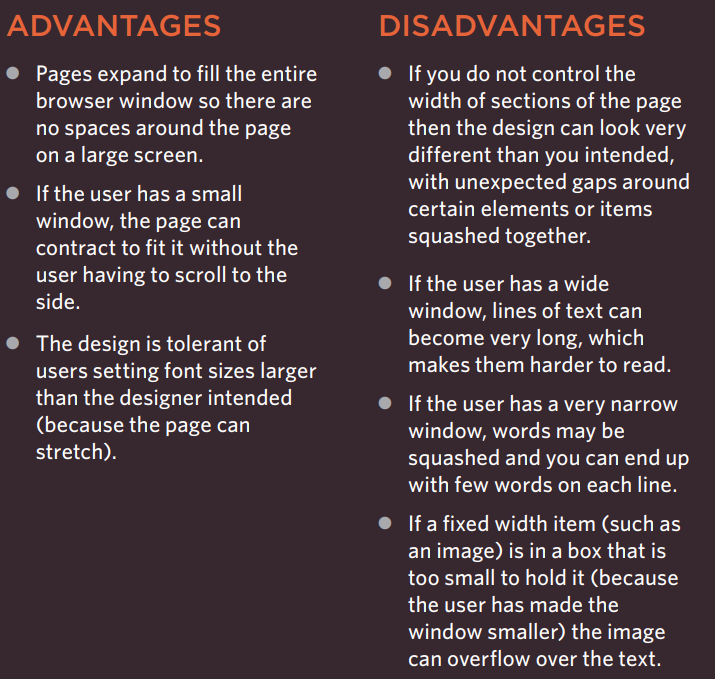
* Fixed width layout designs do not change size as the user increases or decreases the size of their browser window



## Liquid layouts

* Liquid layout designs stretch and contract as the user increases or decreases the size of their browser window.





## A liquid layout

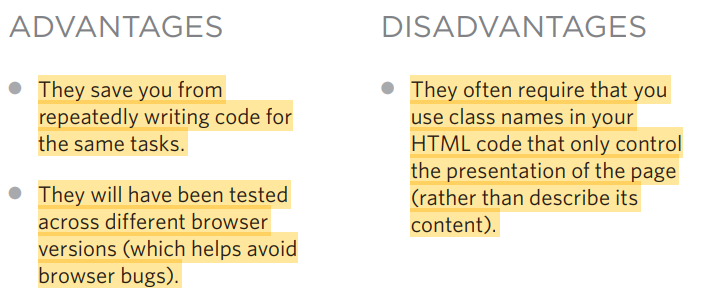
* The liquid layout uses percentages to specify the width of each box so that the design will stretch to fit the size of the screen.

## Layout grids

* Many designers use a grid structure to help them position items on a page, and the same is true for web designers.
* Grids set consistent proportions and spaces between items which helps to create a professional looking design.

## CSS Frameworks

* CSS frameworks aim to make your life easier by providing the code for common tasks, such as creating layout grids, styling forms, creating printer-friendly versions of pages and so on. You can include the CSS framework code in your projects rather than writing the CSS from scratch.



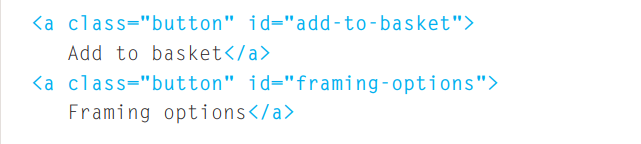
# Chapter 16: Images

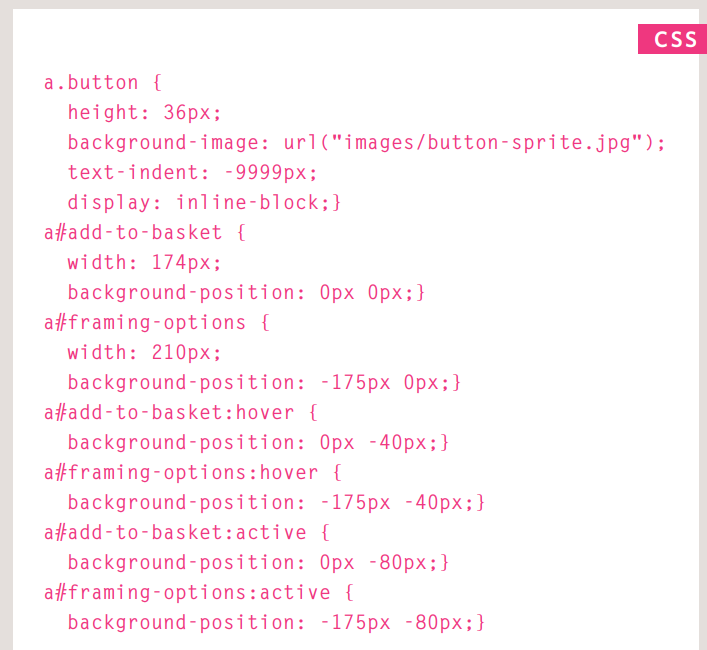
## Controlling sizes of images in CSS

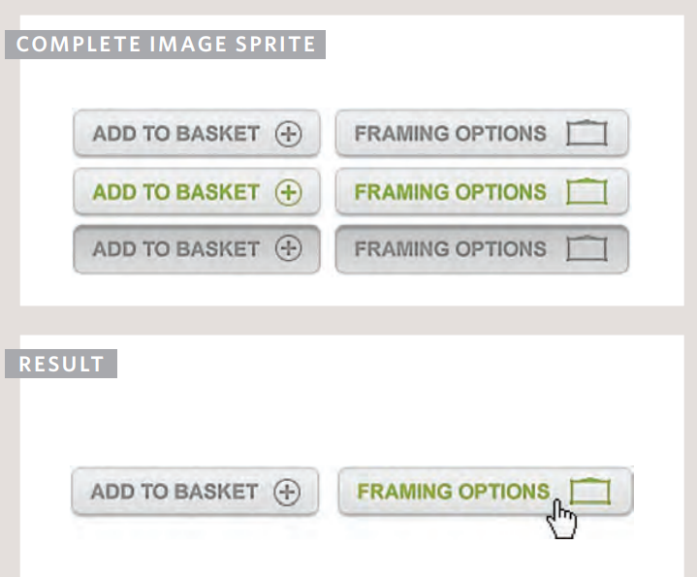
* Specifying image sizes helps pages to load more smoothly because the HTML and CSS code will often load before the images, and telling the browser how much space to leave for an image allows it to render the rest of the page without waiting for the image to download.
* You might think that your site is likely to have images of all different sizes, but a lot of sites use the same sized image across many of their pages.

## Image rollovers & sprites

* When a single image is used for several different parts of an interface, it is known as a sprite.
* The advantage of using sprites is that the web browser only needs to request one image rather than many images, which can make the web page load faster.







# Chapter 17: HTML5 Layout

## HTML5 layout elements

* HTML5 is introducing a new set of elements that help define the structure of a page.

## New HTML5 layout elements

* HTML5 introduces a new set of elements that allow you to divide up the parts of a page. The names of these elements indicate the kind of content you will find in them. They are still subject to change, but that has not stopped many web page authors using them already.

## Headers & Footers

* <header><footer>
* The <header> and <footer> elements can be used for:
* The main header or footer that appears at the top or bottom of every page on the site.
* A header or footer for an individual or within the page.

## Articles

* The <articles> element acts as a container for any section of a page that could stand alone and potentially be syndicated.
* This could be an individual article or blog entry, a comment or forum post, or any other independent piece of content.

## Asides

* The <aside> element has two purposes, depending on whether it is inside an <article> element or not.
* When the <article> element is used inside an <article> element, it should contain information that is related to the article but not essential to its overall meaning
* When the <aside> element is used outside of an <article> element, it acts as a container for content that is related to the entire page.

## Sections

* The <section> element groups related content together, and typically each section would have its own heading
* Because the <section> element groups related items together, it may contain several distinct <article> elements that have a common theme or purpose.
* Alternatively, if you have a page with a long article, the <section> element can be used to split the article up into separate sections.
* The <section> element should not be used as a wrapper for the entire page

## Heading groups <hgroup>

* The purpose of the <hgroup> element is to group together a set of one or more <h1> through <h6> elements so that they are treated as one single heading.

## Figures <figure> <figcaption>

* It is important to note that the article should still make sense
* The <figure> element should also contain a <figcaption> element which provides a text decription for the content of the <figure> element.

## Helping older browsers understand

* Older browsers that do not know the new HTML5 elements
* Also, IE9 was the first version of Internet Explorer to allow CSS rules to be associated with these new HTML5 layout elements. In order to style these elements using earlier versions of IE, you need to use a simple JavaScript known as the HTML5 shiv or HTML5 shim.
* It should be placed inside a conditional comment which checks if the browser version is less than (hence the lt) IE9.

# Chapter 18: Process & Design

## Wireframes

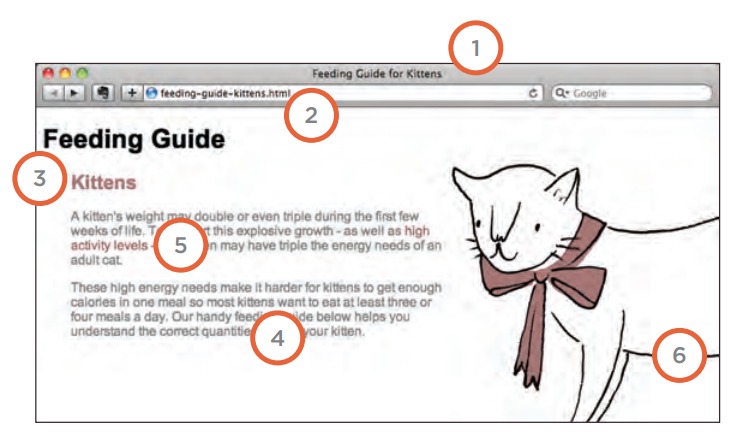
* A wireframe is a simple sketch of the key information that needs to go on each page of a site. It shows the hierarchy of the information and how much space it might require.
* It should focus on what information needs to be on each page and create a visual hierarchy to indicate the most important parts of each page.
* Site maps allow you to plan the structure of a site.
* Wireframes allow you to organize the information that will need to go on each page.

# Chapter 19: Practical information

## Search engine optimization (SEO)

* SEO is a huge topic and several books have been written on the subject. The following pages will help you understand the key concepts so you can improve your website's visibility on search engines.
* The basics
* Search engine optimization (or SEO) is the practice of trying to help your site appear nearer the top of search engine results when people look for the topics that your website covers.
* At the heart of SEO is the idea of working out which terms people are likely to enter into a search engine to find your site and then using these terms in the right places on your site to increase the chances that search engines will show a link to your site in their results
* SEO is often split into two areas: on-page techniques and off-page techniques.
* On-page techniques
* On-page techniques are the methods you can use on your web pages to improve their rating in search engines.
* The main component of this is looking at keywords that people are likely to enter into a search engine if they wanted to find your site, and then including these in the text and HTML code for your site in order to help the search engines know that your site covers these topics.
* Off-page techniques
* Getting other sites to link to you is just as important as on-page techniques. Search engines help determine how to rank your site by looking at the number of other sites that link to yours.

## On-page SEO



1. Page title
2. URL/ WEB address
3. Headings
4. Text
5. Link text
6. Image alt text
7. Page descriptions

## How to identify keywords and phrases

* Determining which keywords to use on your site can be one of the hardest tasks when you start to think about SEO

1. Brainstorm
2. Organize
3. Research
4. Compare
5. Refine
6. Map

## Analytics: Learning about your visitors

* As soon as people start coming to your site, you can start analyzing how they found it, what they were looking at and at what point they are leaving. One of the best tools for doing this is a free service offered by Google called Google Analytics.

## FTP & Third party tools

* To transfer your code and images from your computer to your hosting company, you use something known as File Transfer Protocol.
* Search engine optimization helps visitors find your sites when using search engines.
* FTP programs allow you to transfer files from your local computer to your web server.