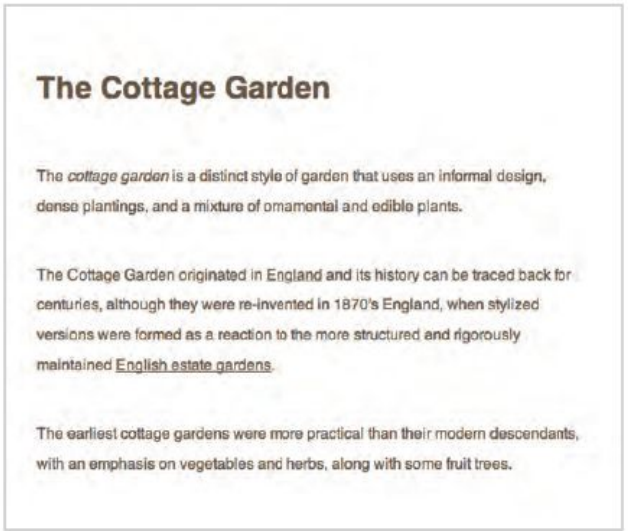
CSS Basics

Introduction and overview of CSS

Understanding CSS:

The key to understanding how CSS works is to imagine that there is an invisible box around every HTML element.



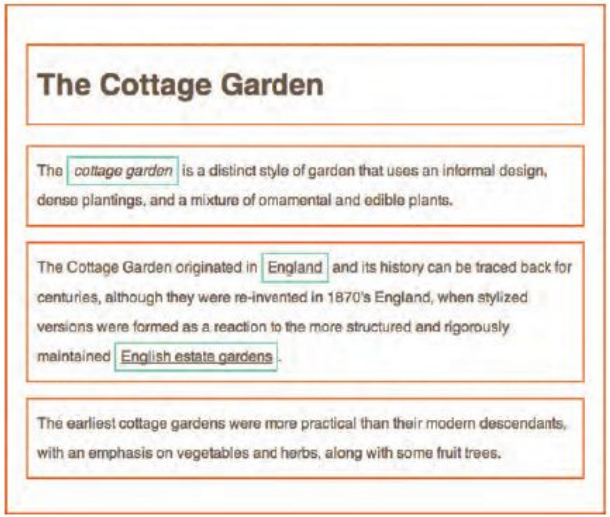
Remember - BLOCK & INLINE ELEMENTS

Block level elements look like they start on a new line. Examples include the <h1>-<h6>, <p> and elements.

Inline elements flow within the text and do not start on a new line. Examples include <b>, <i>, <img>, and .

Understanding interpretation of CSS by browser

On the Image below, you can see the same HTML page, but I have added outlines to each of the elements so that you can see how CSS will treat each element as if it lives inside its own box.



CSS allows you to create rules that control the way that each individual box (and the contents of that box) is presented.

In this example, block level elements are shown with red borders, and inline elements have green borders.

The <body> element creates the first box, then the <h1>, <h2>, <p>, <i>, and <a> elements each create their own boxes within it.

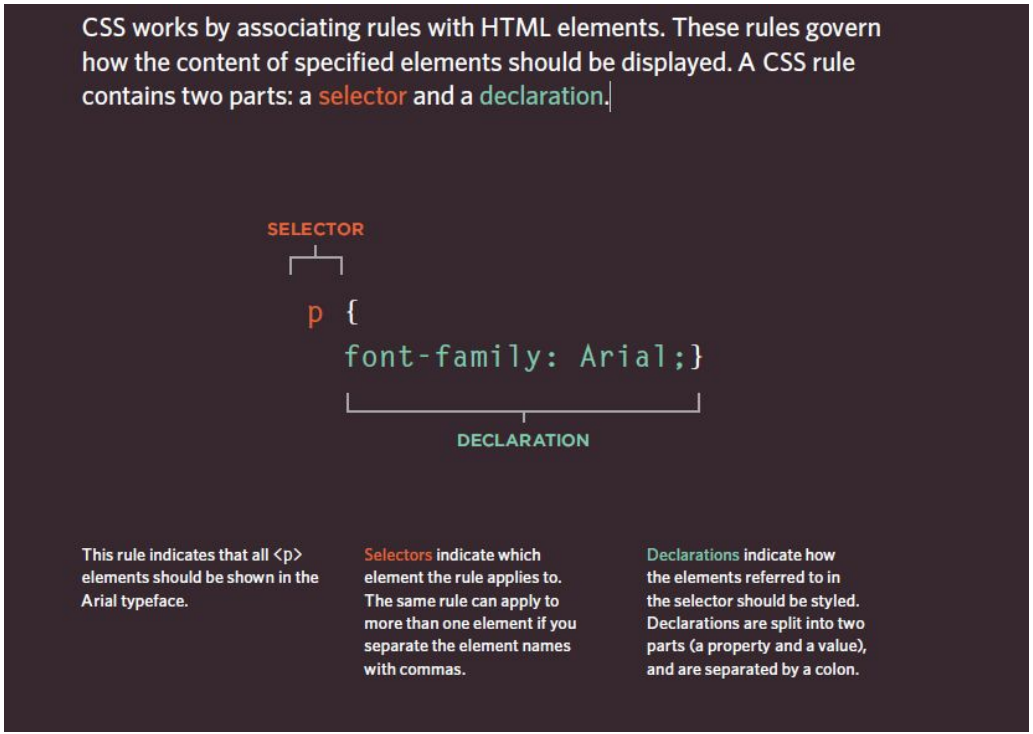
Using CSS, you could add a border around any of the boxes, specify its width and height, or add a background color. You could also control text inside a box — for example, its color, size, and the typeface used.

BOXES:
Width and height
Borders (color, width, and style)
Background color and images
Position in the browser window.

TEXT:
Typeface
Size
Color
Italics, bold, uppercase, lowercase, small-caps

SPECIFIC:
There are also specific ways in which you can style certain elements such as lists, tables, and forms.

CSS Adds Style rules to HTML elements



Each style rule consists of a "selector" and "declarations" of property-value pairs:

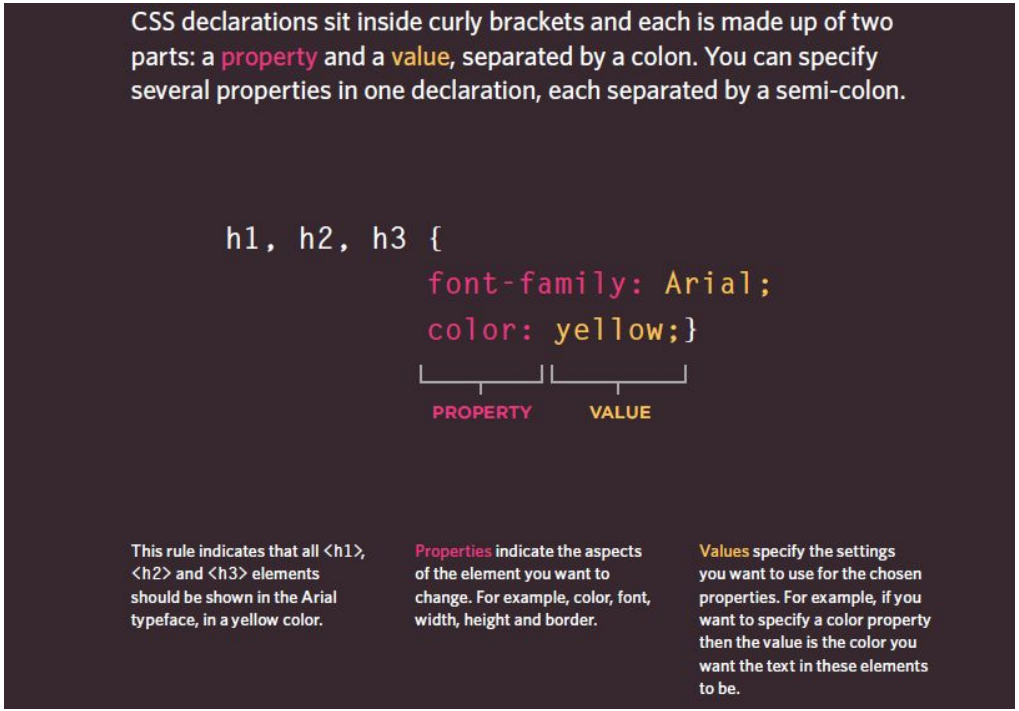
```
selector {
  property: value;
  property: value;
}
```

Declaration

Eg:

```
body {
  color: yellow;
  background-color: black;
}
```

CSS Properties Affect How Elements Are Displayed



Ways to include CSS Styles to HTML

1. External Style Sheet

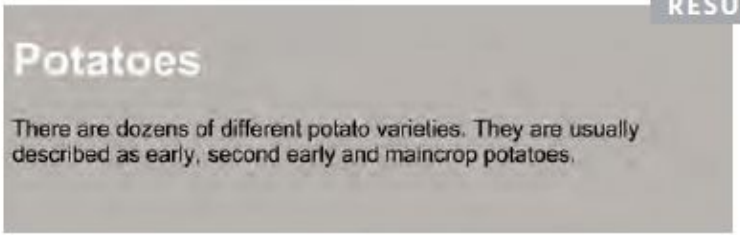
using-external-css.html — HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>Using External CSS</title>
    <link href="css/styles.css" type="text/css"
      rel="stylesheet" />
  </head>
  <body>
    <h1>Potatoes</h1>
    <p>There are dozens of different potato
       varieties. They are usually described as
       early, second early and maincrop.</p>
  </body>
</html>
```

styles.css — CSS

```
body {
    font-family: arial;
    background-color: rgb(185,179,175);}
h1 {
    color: rgb(255,255,255);}
```

RESULT



1. External Style Sheet

<link> : The <link> element can be used in an HTML document to tell the browser where to find the CSS file used to style the page. It is an empty element (meaning it does not need a closing tag), and it lives inside the <head> element. It should use three attributes:

href: This specifies the path to the CSS file (which is often placed in a folder called css or styles).

type: This attribute specifies the type of document being linked to. The value should be text/css.

rel: This specifies the relationship between the HTML page and the file it is linked to. The value should be stylesheet when linking to a CSS file.

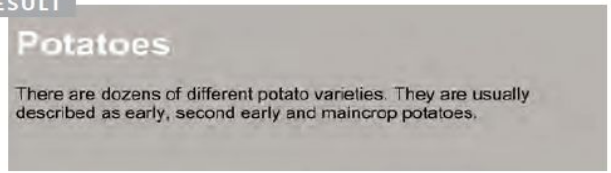
An HTML page can use more than one CSS style sheet. To do this it could have a <link> element for every CSS file it uses. For example, some authors use one CSS file to control the presentation (such as fonts and colors) and a second to control the layout.

2. Internal CSS

HTML + CSS using-internal-css.html

```
<!DOCTYPE html>
<html>
  <head>
    <title>Using Internal CSS</title>
    <style type="text/css">
      body {
          font-family: arial;
          background-color: rgb(185,179,175);}
      h1 {
          color: rgb(255,255,255);}
    </style>
  </head>
  <body>
    <h1>Potatoes</h1>
    <p>There are dozens of different potato
       varieties. They are usually described as
       early, second early and maincrop.</p>
  </body>
</html>
```

RESULT



<style>

You can also include CSS rules within an HTML page by placing them inside a **<style> element**, which usually sits **inside the <head> element of the page.**

The <style> element should use the type attribute to indicate that the styles are specified in CSS. The value should be text/css.

When building a site with more than one page, you should use an external CSS style sheet.

This:

- Allows all pages to use the same style rules (rather than repeating them in each page).
- Keeps the content separate from how the page looks.
- Means you can change the styles used across all pages by altering just one file (rather than each individual page).

Inline CSS

CSS styles can be assigned to individual HTML elements via the style attribute:

<tag **style="property1: value; property2: value;"**> </tag>

<h1 **style="color:red; text-decoration: underline;"**>Hello</h1>

Inline styles will have precedence over all other styles.

Inline CSS should be avoided as it increases maintenance cost and blurs line between presentation and content.

CSS Selectors

There are many different types of CSS selector that allow you to target rules to specific elements in an HTML document.

CSS selectors are case sensitive, so they must match element names and attribute values exactly.

css-selectors.html HTML

```
<!DOCTYPE html>
<html>
  <head>
    <title>CSS Selectors</title>
  </head>
  <body>
    <h1 id="top">Kitchen Garden Calendar</h1>
    <p id="introduction">Here you can read our
       handy guide about what to do when.</p>
    <h2>Spring</h2>
    <ul>
      <li><a href="mulch.html">
             Spring mulch vegetable beds</a></li>
      <li><a href="potato.html">
             Plant out early potatoes</a></li>
      <li><a href="tomato.html">
             Sow tomato seeds</a></li>
      <li><a href="beet.html">
             Sow beet seeds</a></li>
      <li><a href="zucchini.html">
             Sow zucchini seeds</a></li>
      <li><a href="rhubarb.html">
             Deadhead rhubarb flowers</a></li>
    </ul>
    <p class="note">
       This page was written by
      <a href="mailto:ivy@example.org">
         ivy@example.org</a> for
      <a href="http://www.example.org">Example</a>.
    </p>
    <p>
      <a href="#top">Top of page</a>
    </p>
  </body>
</html>
```

SELECTOR	MEANING	EXAMPLE
UNIVERSAL SELECTOR	Applies to all elements in the document	`* {}` Targets all elements on the page
TYPE SELECTOR	Matches element names	`h1, h2, h3 {}` Targets the <h1>, <h2> and <h3> elements

CLASS SELECTOR	Matches an element whose `class` attribute has a value that matches the one specified after the period (or full stop) symbol	`.note {}`
		Targets any element whose `class` attribute has a value of `note`
		`p.note {}`
		Targets only <p> elements whose `class` attribute has a value of `note`

ID SELECTOR	Matches an element whose `id` attribute has a value that matches the one specified after the pound or hash symbol	`#introduction {}` Targets the element whose `id` attribute has a value of `introduction`
CHILD SELECTOR	Matches an element that is a direct child of another	`li>a {}` Targets any <a> elements that are children of an element (but not other <a> elements in the page)
DESCENDANT SELECTOR	Matches an element that is a descendent of another specified element (not just a direct child of that element)	`p a {}` Targets any <a> elements that sit inside a <p> element, even if there are other elements nested between them
ADJACENT SIBLING SELECTOR	Matches an element that is the next sibling of another	`h1+p {}` Targets the first <p> element after any <h1> element (but not other <p> elements)
GENERAL SIBLING SELECTOR	Matches an element that is a sibling of another, although it does not have to be the directly preceding element	`h1~p {}` If you had two <p> elements that are siblings of an <h1> element, this rule would apply to both

Common Selectors

```
selector {
        property: values;
    }
```

The selector is used to select which elements in the HTML page will be given the styles inside the curly braces.

Types of Selectors

1. element
2. id
3. class
4. position in document

CSS Selector Types

Types of Selectors: element

```
p {      }
```

Selects all p elements in the entire document.

Types of Selectors: id

```
#header {    }
```

Selects any element with the id “header", e.g.

```
<p id="header"></p>
```

Element ids are unique, so that should only be one element.

The "#" is how you tell CSS "this is an id."

CSS Selectors Cont..

Types of Selectors: class

```
.warning {
  color: red;
}
```

Selects any element with the class name “warning”, e.g.

```
<p class="warning"></p>
```

Multiple elements can have the same class name.

The "." is how you tell CSS "this is a class name."

Types of Selectors: class

An element can have only 1 id but multiple classes, so you can take advantage of that for greater flexibility.

```
.intro {
  background-color: blue;
}
.desc {
  color: red;
}
```

```
<p class="desc intro">hi</span> -> hi
<p class="desc">hi</span> -> hi
<p class="intro">hi</span> -> hi
```

CSS Selector: Position in Document

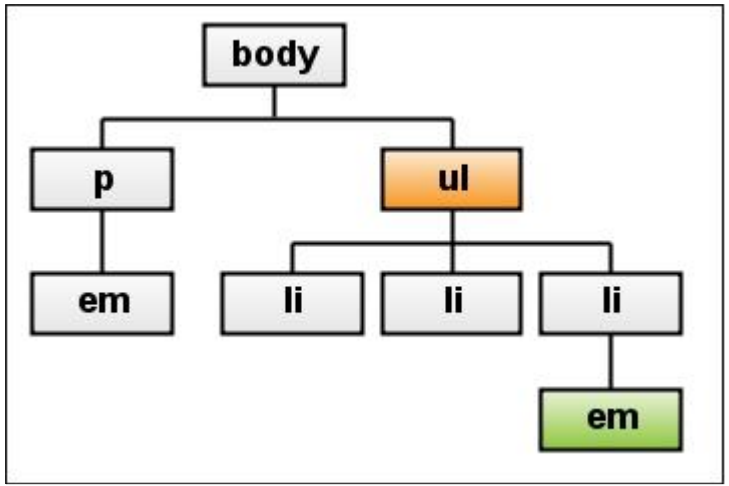
Types of selectors: position in document

```
ul em { color: yellow; }
```

Selects any em element that's a descendant of a “**ul”** element.

The " " (space) is how you say "find a descendant."

```
<body>
    <ul>
        <li><em>Hi</em></li>
    </ul>
    <p><em>Bye</em></p>
</body>
```



CSS Selector: id+position, element+class

Types of selectors: id + position

```
#related-brands li {
  color: gray;
}
```

Selects any <li> element that is a descendant of any element with an id that equals "related-brands."

```
<ul id="related-brands">
        <li>Rice Krispies </li>
        <li>NutriGrain <li>
</ul>
```

Types of selectors: element + class

```
li.special { color: yellow; }
```

Selects any li element with a class attribute that contains the word "special".

```
<ul>
        <li>Rice Krispies </li>
        <li class="special">NutriGrain </li>
</ul>
```

Warning: If you place a space after the ".", it'll look for descendants of li with class of "special."

CSS Pseudo Selector Class

Types of selectors: pseudo classes

A set of "pseudo classes" can style anchor elements depending on their state.

```
a:link { /* unvisited link */  color: red; }
a:visited { /* visited link */ color: blue; }
a:hover { /* moused over link */ color: green; }
a:active { /* current link */ color: purple; }
a:focus {  /* focused link */ color: purple; }
```

Group Selectors

You can group selectors to apply the same style to all of the selectors by separating them with commas:

```
a:hover, a:active:, a:focus {
  background-color: yellow;
}
```

The selector: Cascade rules

Generally:

- id is more specific than a class, class is more specific than element.
- the longer the selector, the more specific it is
- If style rules are equally specific, the last one wins!

Example:

```
#a #b h1 { color: red; } /* winner! */
#a h1 { color: blue; }
```

Coding Conventions

It is possible to write CSS like this:

```
Selector{property:values} selector{property:values}
```

But it's preferred to write it like this:

```
selector {
  property: values;
}

selector {
  property: values;
}
```

Notice: space between "selector" and "{", 2 spaces before property-value pair, space after "property:", semi-colon after "values", line between rules.

Coding Conventions

Some rules to follow when making IDs and class names:

- Describe the content, not the presentation ("warning", not "redbox").
- Use all lowercase, and hyphens when needed for readability ("header-info", not "headerInfo").
- Use hyphens to show that a class or ID is part of something else. (e.g. "footer", "footer-copyright", and "footer-logo").

Comments will be ignored by the browser and are useful for documenting your styles to other humans or commenting out rules.

```
/* Comment here */
p
{
  margin: 1em; /* Comment here */
  padding: 2em;
  /* color: white; */
  background-color: blue;
}

/*
  multi-line
  comment here
*/
```

CSS Property Value pair

Property Value Pairs

Each property can have one or more comma separated values.

```
font: italic 12px sans-serif;
color: #333;
background-color: red;
font-family: Arial, sans-serif;
```

Properties: color

The "color" property changes the text color. Colors can be specified either by name, for the most common colors, or by hexadecimal value.

```
color: red;
color: #ff0000;
color: rgb(255, 0, 0);
```

This property is inherited, which means it'll also be applied to all descendant elements but can be overridden by more specific rules. This rule makes all the body text red unless specified otherwise:

```
body {
  color: red;
}
```

CSS Property Value pair

Properties: background-color

The "background-color" property changes the background color. Besides the BODY element, all elements default to a transparent background.

```
background-color: black;
background-color: #000000;
background-color: rgb(0,0,0);

body {
  background-color: yellow;
}

table {
  background-color: #FFCC00;
}
```

Property: font-family

The "font-family" property specifies the font family (or "font face") of the text. You can specify either a specific font name or a generic family name (serif, sans-serif, monospace, cursive).

```
font-family: "Times New Roman";
font-family: sans-serif;
```

A comma separated list of font families can be specified if you want the browser to prefer one but use the others as backup options.

```
font-family: "Times New Roman", serif;
font-family: "Arial", sans-serif;
font-family: Courier, monospace;
```

CSS Property Value pair

Property: font-size

The "font-size" property specifies the size of a font. It can be specified as a fixed size in various units, a percentage, or as a predefined keyword.

```
font-size: 1.5em;
font-size: 12px;
font-size: 100%;
font-size: larger;
```

The "em" unit lets you set the size of the text relative to the text around it. This makes the page resize nicely in proportion if the user changes their default font-size. The default size is "1em".

The "px" unit lets you size font in terms of pixels, which is the unit also used to size images and other elements. It is easier to understand than em, but doesn't work as well when printing or resizing.

There are various keywords that can be used if you're not as worried about the precise sizing: xx-small, x-small, small, medium, large, x-large, xx-large.

CSS Property Value pair

Property: font-style

The "font-style" property specifies the font style of the text, either "normal" by default or "italic".

font-style: italic;

Italicized text.

Property: font-weight

The "font-weight" property specifies the thickness of the font. The default is "normal" and the typical override is "bold". You can also specify "bolder", "lighter", or a number from 100 to 900.

font-weight: bold;

Bold text!

Shorthand Property values

A "shorthand" property in CSS lets you specify multiple properties in one property, for conciseness purposes. Instead of specifying each "font-" property separately, you can bundle them up in one "font" property.

```
table.geeky {
  font-weight: bold;
  font-style: italic;
  font-size: 10px;
  font-family: sans-serif;
}
```

Those four rules can be written as:

```
table {
  font: italic bold 10px sans-serif;
}
```

Other text properties

line-height: 10px;

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis aliquam convallis ligula ut suscipit. Phasellus hendrerit, enim scelerisque...

text-align: center;

Centered text.

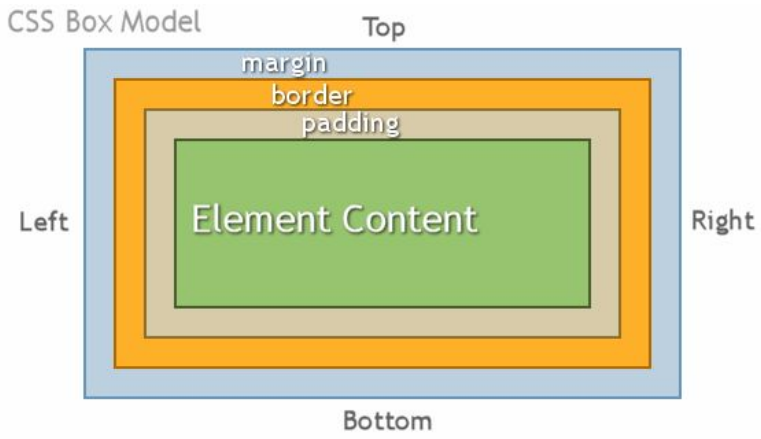
text-decoration: underline;

Underlined text

text-indent: 20px;

Indented text.

Margin vs Padding



- **Padding** is applied to the inside of the border of your element so you can control how far the content is away from the border.
- **Margin** is applied to the outside of the border of your element, so you can control how far the element is away from other elements.

Special usage: if you use a negative margin, you can pull the element closer to other elements. and also good to keep in mind: padding will increase the dimensions of your element, whereas margin won't.

CSS Rules Cascade

If there are two or more rules that apply to the same element, it is important to understand which will take precedence.

LAST RULE

If the two selectors are identical, the latter of the two will take precedence. Here you can see the second **<i>** selector takes precedence over the first.

SPECIFICITY

If one selector is more specific than the others, the more specific rule will take precedence over more general ones. In this

example:
h1 is more specific than *,
p b is more specific than p,
p#intro is more specific than p

IMPORTANT

You can add “!important” after any property value to indicate that it should be considered more important than other rules that apply to the same element.

cascade.html — HTML

```
<h1>Potatoes</h1>
<p id="intro">There are <i>dozens</i> of different
   <b>potato</b> varieties.</p>
<p>They are usually described as early, second early
   and maincrop potatoes.</p>
```

CSS

```
* {
  font-family: Arial, Verdana, sans-serif;}
h1 {
  font-family: "Courier New", monospace;}
i {
  color: green;}
i {
  color: red;}
b {
  color: pink;}
p b {
  color: blue !important;}
p b {
  color: violet;}
p#intro {
  font-size: 100%;}
p {
  font-size: 75%;}
```

RESULT

Potatoes

There are *dozens* of different potato varieties.

They are usually described as early, second early and maincrop potatoes.

Inheritance

If you specify the **font-family or color properties** on the **<body>** element, they will apply to most child elements. This is because the value of the font-family property is inherited by child elements.

You can compare this with the background-color or border properties; they are not inherited by child elements.

You can force a lot of properties to inherit values from their parent elements by using inherit for the value of the properties.

In this example, the element with a class called page inherits the padding size from the CSS rule that applies to the <body> element.

HTML inheritance.html

```
<div class="page">
  <h1>Potatoes</h1>
  <p>There are dozens of different potato
     varieties.</p>
  <p>They are usually described as early, second
     early and maincrop potatoes.</p>
</div>
```

CSS

```
body {
  font-family: Arial, Verdana, sans-serif;
  color: #665544;
  padding: 10px;}
.page {
  border: 1px solid #665544;
  background-color: #efefef;
  padding: inherit;}
```

RESULT