

Styling the New Web

Web Usability with Style Sheets

Steven Pemberton

CWI and W3C
Kruislaan 413
1098 SJ Amsterdam
The Netherlands

Steven.Pemberton@cwi.nl
www.cwi.nl/~steven

Agenda

The day is split into four blocks, each of 90 minutes.

Each block consists of about 45 minutes lecture, followed by 45 minutes practical.

The breaks between blocks are 30 minutes, with 90 minutes for lunch.

Block 1

Introduction, basic CSS: selectors, fonts, colours.

Break

Block 2

Intermediate level: advanced selectors, inheritance, margins, borders, padding.

Lunch

Block 3

Advanced: text properties, background, positioning, cascading.

Break

Block 4

The Future: XML and XHTML.

About the Instructor

Steven Pemberton is a researcher at the CWI, The Centre for Mathematics and Computer Science, a nationally-funded research centre in Amsterdam, The Netherlands, the first non-military Internet site in Europe.

Steven's research is in interaction, and how the underlying software architecture can support the user. At the end of the 80's he built a style-sheet based hypertext system called Views.

Steven has been involved with the World Wide Web since the beginning. He organised two workshops at the first World Wide Web Conference in 1994, chaired the first W3C Style Sheets workshop, and the first W3C Internationalisation workshop. He was a member of the CSS Working Group from its start, and is a long-time member (now chair) of the HTML Working Group, and co-chair of the XForms Working Group. He is co-author of (amongst other things) HTML 4, CSS, XHTML and XForms.

Steven is also Editor-in-Chief of ACM/interactions.

Objectives

HTML has for too long, and incorrectly, been seen as a purely presentation language. It was originally designed as a structure description language, but extra elements were added later by browser manufacturers in order to influence the presentation. This has had the effect of limiting Web site usability by introducing presentation elements that slow down Web access, reduce or prevent accessibility to the sight-impaired, limit the sorts of devices that may be used to view websites, and reduce the end-user's options when viewing a document.

The World Wide Web Consortium (W3C) started the Style Sheet activity in 1995 in order to get HTML back to its pure form. The result of this was Cascading Style Sheets (CSS), which allows the separation of content and presentation in Web sites.

Using style sheets has many benefits, including:

- Separation of content and presentation means that Web pages are easier to write.
- Since images are no longer needed to represent styled text, Web pages download significantly faster.
- By separating out the presentation elements, blind and other sight-impaired users are able to access the Web much more easily, especially since CSS explicitly supports aural browsers.
- By allowing style sheets to specify sizes in relation to other sizes, rather than as absolute sizes, people with reduced sight can scale pages up and still see them as they were intended.
- By not coding device-dependent presentations in the HTML pages are viewable on a wider range of devices.
- You can now design the look of your site in one place, so that if you change your house style, you only need to change one file to update your entire site.

Even if the Web remained based on HTML, these would be enough reasons to use style sheets. However, the Web is now going in a new direction: XML, and XML has no inherent presentation semantics at all. To use XML you *have* to use a style sheet to make your site visible.

As a part of the movement to XML, a new version of HTML, called XHTML, is being developed. Since all presentation-oriented elements are being dropped, style sheets will become essential there too.

So the objectives of this course are to give an advanced introduction to the use of CSS to style HTML and XML documents, showing in passing how this can improve usability, and to give an introduction to the use of XML and XHTML.

Although most attention will be paid to CSS level 1, since this is the version currently most widely implemented, CSS 2 will also be treated, and information about what can be expected in CSS 3 will be given.

These notes have been produced entirely in XHTML and CSS, using different stylesheets for printing, screen use, and presentation. In particular, there are only three images used (to demonstrate the use of images): a star, and two versions of a flower. All other things that look like images are in fact just styled with CSS.

It should go without saying that to properly appreciate this document, you have to view it with a browser with good CSS support.

1: Course Plan

- Part 1
Introduction, basic CSS: selectors, fonts, colours; Practical
- Part 2
Intermediate-level stuff: advanced selectors, inheritance, margins, borders, padding; Practical
- Part 3
Advanced stuff: text properties, background, positioning; Practical
- Part 4
The Future: XML and XHTML; Practical

2: HTML and SGML

- HTML (up to now) has been an SGML application.
- SGML is intended to define the structure of documents
For instance, `<H1> </H1>` defines a heading without specifying how it should look.

` ... `

specifies a list of items.
- These classifications often have semantic significance. `<I>` and `` were mistakes, use `` and `` instead

3: Contamination

- Netscape started to add their own tags, based on the idea that with their market penetration they could get a head start.
- Unfortunately most tags added by Netscape are presentation-oriented tags – which do not fit in the structure orientation of standard HTML – such as <BLINK> and
- These do specify how the item should look, and have no inherent semantics; Microsoft also followed suit.

4: Style Sheets

- In order to get HTML back to being a structure language, W3C hosts work on Style Sheets, and producing a Style Sheet Language CSS – Cascading Style Sheets.
- Aims:
 - easy to write
 - easy to implement
 - has a development path.
- CSS is a 90% solution
- For all typesetting possibilities XSL is being developed

5: CSS

- CSS is a language that allows you to specify how a document, or set of documents, should look.
- Advantages:
 - Separates content from presentation
 - Makes HTML a structure language again
 - Makes HTML easier to write (and read)
 - All HTML styles (and more) are possible
 - You can define a house style in one file
 - Accessible for the sight-impaired
 - You can still see the page on non-CSS browsers
- CSS is also an enabling technology for XML

6: Motivation

Csszengarden.com is a website with essentially [one HTML page](#).

And hundreds of beautiful, breathtaking CSS stylesheets applied to that one page.

css Zen Garden

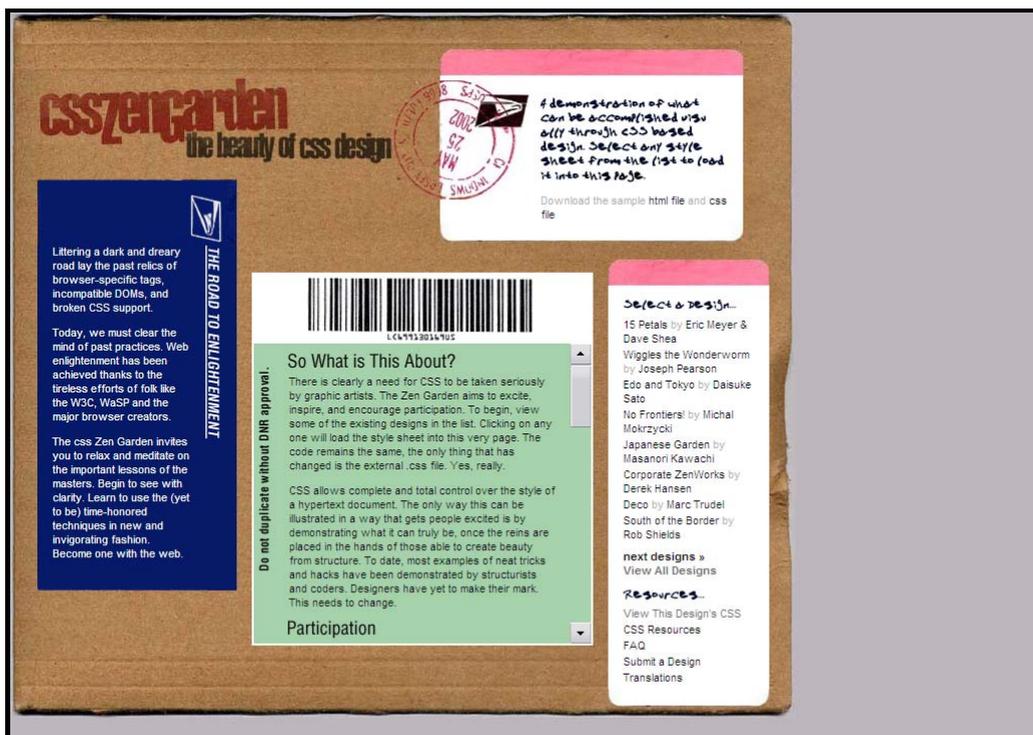
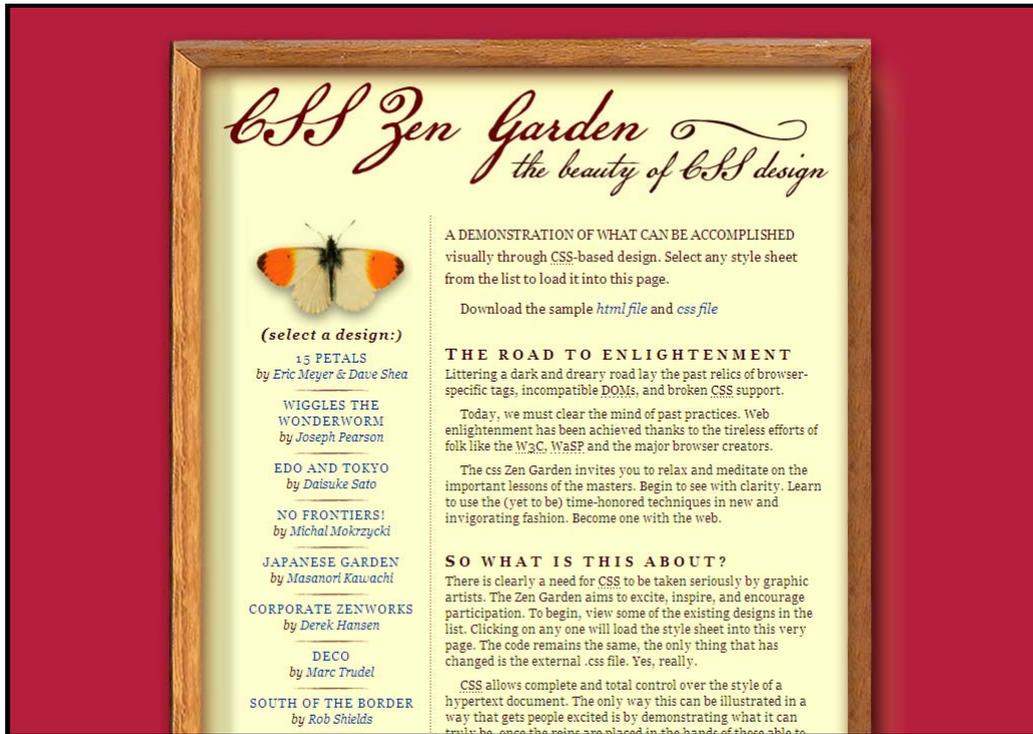
The Beauty of CSS Design

A demonstration of what can be accomplished visually through CSS-based design. Select any style sheet from the list to load it into this page.

Download the sample [html file](#) and [css file](#)

It uses very simple HTML: div, h1, h2, h3, span, p, ul/li, a, acronym

As you look at each presented page, you have to keep repeating to yourself: this is the same HTML page.





A demonstration of what can be accomplished visually through CSS-based design. Select any style sheet from the list to load it into this page.
Download the sample [html file](#) and [css file](#).

THE ROAD TO ENLIGHTENMENT

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, and broken CSS support.

Today, we must clear the mind of past practices. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP and the major browser creators.

The css Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the (yet to be) time-honored techniques in new and invigorating fashion. Become one with the web.

SELECT A DESIGN:

- 15 Petals by ERIC MEYER & DAVE SHEA
- Wiggles the Wonderworm by JOSEPH PEARSON
- Edo and Tokyo by DAISUKE SATO
- No Frontiers! by MICHAL MOKRZYCKI
- Japanese Garden by MASANORI KAWACHI
- Corporate ZenWorks by DEREK HANSEN
- Deco by MARC TRUDEL
- South of the Border by ROB SHIELDS

ARCHIVES:

→ next designs → View All Designs

RESOURCES:

→ css: The Designer's CSS → CSS Resources

→ FAQ → Submit a Design → Translations

SO WHAT IS THIS ABOUT?

There is clearly a need for CSS to be taken seriously by graphic artists. The Zen Garden aims to excite, inspire, and encourage participation. To begin, view some of the existing designs in the list. Clicking on any one will load the style sheet into this very page. The code remains the same, the only thing that has changed is the external .css file. Yes, really.

CSS allows complete and total control over the style of a hypertext document. The only way this can be illustrated in a way that gets people excited is by demonstrating what it can truly be, once the reins are placed in the hands of those able to create beauty from structure. To date, most examples of neat tricks and hacks have been demonstrated by structurists and coders. Designers have yet to make their mark. This needs to change.

Erase Image (s)

Yes / No ?

XHTML CSS CC 508 AAA

PLACE

CITY: Alicante
EVENT: Hogueras de San Juan
DATE: 20/06/2003

Focus



zen garden

The road to enlightenment

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, and broken CSS support.

Today, we must clear the mind of past practices. Web enlightenment has been achieved thanks to the tireless efforts of folk like the W3C, WaSP and the major browser creators.

The css Zen Garden invites you to relax and meditate on the important lessons of the masters. Begin to see with clarity. Learn to use the (yet to be) time-honored techniques in new and invigorating fashion. Become one with the web.

so what is this about ?

There is clearly a need for CSS to be taken seriously by graphic artists. The Zen Garden aims to excite, inspire, and encourage participation. To begin, view some of the existing designs in the list. Clicking on any one will load the style sheet into this very page. The code remains the same, the only thing that has changed is the external .css file. Yes, really.

CSS allows complete and total control over the style of a hypertext document. The only way this can be illustrated in a way that gets people excited is by demonstrating what it can truly be, once the reins are placed in the hands of those able to create beauty from structure. To date, most examples of neat tricks and hacks have been demonstrated by structurists and coders. Designers have yet to make their mark. This needs to change.

SELECT A DESIGN:

- 15 Petals by Eric Meyer & Dave Shea
- Wiggles the Wonderworm by Joseph Pearson
- Edo and Tokyo by Daisuke Sato
- No Frontiers! by Michal Mokrzycki
- Japanese Garden by Masanori Kawachi
- Corporate ZenWorks by Derek Hansen
- Deco by Marc Trudel
- South of the Border by Rob Shields

ARCHIVE

RESOURCES

A demonstration of what can be accomplished visually through CSS-based design. Select any style sheet from the list to load it into this page.
Download the sample [html file](#) and [css file](#).

dutchcult





THE BEAUTY OF CSS DESIGN

A DEMONSTRATION OF WHAT CAN BE ACCOMPLISHED VISUALLY THROUGH CSS-BASED DESIGN. SELECT ANY STYLE SHEET FROM THE LIST TO LOAD IT INTO THIS PAGE.

DOWNLOAD THE SAMPLE HTML FILE AND CSS FILE

THE ROAD TO ENLIGHTENMENT

LITTERING A DARK AND DREARY ROAD LAY THE PAST RELICS OF BROWSER-SPECIFIC TAGS, INCOMPATIBLE DOMS, AND BROKEN CSS SUPPORT.

TODAY, WE MUST CLEAR THE MIND OF PAST PRACTICES. WEB ENLIGHTENMENT HAS BEEN ACHIEVED THANKS TO THE TIRELESS EFFORTS OF FOLK

15 PETALS

BY ERIC MEYER & DAVE SHEA

WIGGLES THE WONDERWORM

BY JOSEPH PEARSON

EDO AND TOKYO

BY DAISUKE SATO

NO FRONTIERS!

BY MICHAL MOKRZYCKI

JAPANESE

GARDEN

BY MASANORI KAWACHI

CSS ZEN GARDEN PRESENTS

A DEMONSTRATION OF WHAT CAN BE ACCOMPLISHED VISUALLY THROUGH CSS-BASED DESIGN. SELECT ANY STYLE SHEET FROM THE LIST TO LOAD IT INTO THIS PAGE.

DOWNLOAD THE SAMPLE HTML FILE AND CSS FILE

STARRING
IN ORDER OF APPEARANCE

INVASION OF THE BODY SWITCHERS
by ANDY CLARKE

GOLDEN CUT
by PETR STANICEK

THE HALL
by MICHAEL SIMMONS

NEAT & TIDY
by OLIVIA DALE

CUBE GARDEN
by MASANORI KAWACHI

DJ STYLE
by RAMON BISPO

THE FINAL ENDING
by RAY HENRY

CONTEMPORARY NOUVEAU
by DAVID HELLSING

INVASION OF THE BODY SWITCHERS



THE ROAD TO
ENLIGHTENMENT

THE CSS ZEN GARDEN INVITES YOU TO RELAX AND MEDITATE ON THE IMPORTANT LESSONS OF THE MASTERS. BEGIN TO SEE WITH CLARITY. LEARN TO USE THE (YET TO BE) TIME-HONORED TECHNIQUES IN NEW AND INVIGORATING FASHION. BECOME ONE WITH THE WEB.



7: Separating Content and Presentation: Author Advantages

- Easier to write your documents
- Easier to change your documents
- Easy to change the look of your documents
- Access to professional designs
- Your documents are smaller
- Visible on more devices
- Visible to more people

8: Separating Content and Presentation: Webmaster Advantages

- Separation of concerns
- Simpler HTML, less training
- Cheaper to produce, easier to manage
- Easy to change house style
- Documents are smaller: less bandwidth
(ESPN claims that CSS saves them 2 Tbytes *per day!*)
- Reach more people
- Search engines find your stuff easier
- Visible on more devices

9: Separating Content and Presentation: Reader Advantages

- Faster download (one of the top 4 reasons for liking a site)
- Easier to find information
- You can actually read the information if you are sight-impaired
- Information more accessible
- You can use more devices

10: Separating Content and Presentation: Implementor Advantages

- Improves the implementation (separation of concerns)
- Can produce smaller browsers

11: Levels

- CSS has been designed with upwards and downwards compatibility in mind.
 - CSS1: basic formatting, fonts, colours, layout; quick and easy to implement
 - CSS2: more advanced formatting; aural style sheets
 - CSS3: printing, multi-column, ...
- In general a valid CSS1 style sheet is also a valid CSS2 style sheet.
- In general a CSS2 style sheet can be read and used by a CSS1-supporting browser.

12: Check your log files!

- More than 95% of surfers now use a CSS1-compatible browser:
 - Microsoft IE 3, 4, 5, 6
 - Netscape 4, 6, 7, 8 and its Mozilla-based relatives
 - Opera 3.5, 4, 5, 6, 7, 8
- While the quality of the support for CSS on these browsers varies, though is continually improving, you never need to use the tag again!
- Today we'll be largely talking about CSS1, since it is widely implemented

13: What is the most important property of a website?

Forrester did research among 8000+ users on why they chose one website above another equivalent one. Reasons were:

- Content 75%
- Usability 66%
- Speed 58%
- Freshness 54%

All other reasons were 14% or lower.

CSS can't help you with Content or Freshness, but it can with the other two!

14: Why is CSS good for usability?

- Presentation is not hard-wired in the HTML
- Users can make their own choices (font size, colours, etc), and override the documents
- Pages load faster
- Pages become more accessible for the sight-impaired (who can use speech browsers)
- Pages are viewable on a wider range of platform types

15: Technical stuff

So much for the motivation, now on to the technical part

16: Using CSS

Normally, you put your CSS descriptions in an external file, and link to that from your HTML:

```
<html>
  <head>
    <link rel="stylesheet"
          type="text/css"
          href="your-filename.css">
  </head>
  <body> ...</body>
</html>
```

17: Inline style is also possible

You can also put your style sheets in the head of your HTML document:

```
<head>
  <style type="text/css">
    h1 { color: blue }
  </style>
</head>
```

For many reasons, it is better to use external style sheets

18: Style sheets for XML

For XML use a processing instruction:

```
<?xml-stylesheet type="text/css"
  href="your-filename.css"?>
```

Put before first element of the document

19: HTML Style Attributes

HTML also allows you to use a STYLE attribute:

```
<P STYLE="color: red">Stop!</P>
```

This is bad practice, and undoes many of the advantages of CSS.

Doesn't (necessarily) work for XML.

20: Warning about HTML: <p>

- <P> is not the same as
!
- Don't do this:

```
<H1>The Title</H1>
This is the first paragraph<P>
And this is the second
```

- But this:

```
<H1>The Title</H1>
<P>This is the first paragraph</P>
<P>And this is the second</P>
```

- **Your CSS will work better, and new versions of HTML require it anyway.**

21: Warning about old browsers

CSS implementations are now quite good, but older browsers had a variety of mistakes. Unfortunately, some browser manufacturers want to offer backwards compatibility with those buggy old browsers. So they have two modes: compliant mode, and legacy mode.

To decide which mode to use they look at the document.

22: Legacy and compliant modes

- For XML it is always in compliant mode
- For HTML it is compliant mode if you include a DOCTYPE at the top of the document, for instance:

```
<!DOCTYPE HTML
PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

So if you want your CSS to work right, always include a DOCTYPE at the head of your document.

23: About Valid Documents

HTML was designed so that incorrect documents are corrected by the browser.

However, you can never know exactly how the browser has corrected your document. This means that you don't know the exact structure of your underlying document.

Which means that you can't be sure how your CSS will be applied.

Therefore: always ensure you have a valid document.

For instance, check it at the W3C validator at validator.w3.org

24: Structure of CSS1

- CSS has rules consisting of selectors and blocks;
- Blocks are a series of declarations between curly brackets, separated by semicolons:

```
H1 { font-family: helvetica, sans-serif;  
      font-weight: bold }
```

- Declarations consist of a property followed by a value (declarations may also be empty):

```
font-size: 10pt
```

25: Comments

- Comments are expressed between `/*` and `*/`
- Example:

```
/* This is a comment */
```

26: Basic Selectors

- Basic selectors are just element names
 - H1
 - BODY
 - P
- Several rules can be joined together using the comma:
 - H1, H2, H3, H4, H5, H6 { font-family: helvetica, sans-serif}
- **Since not all browsers accept it, don't use *html* as a selector, except to remove styling the browser added: use *body* instead**

27: Warning about HTML: case

- HTML is case-insensitive. You can write

```
<H1>Title</H1>
or
<h1>Title</h1>
```
- Therefore, you can write selectors either as

```
H1 {font-weight: bold}
or
h1 {font-weight: bold}
```
- **Newer versions of HTML are case sensitive, so specify your selectors in lower case**

28: Examples

- h1, h2, h3 { font-family: helvetica, sans-serif }
- body { color: white; background-color: black }
- p { text-align: justify }

29: All matching rules apply

You can have as many rules as you like for a selector. All rules that apply to an element get applied. Clashing declarations are resolved by giving priority to more specific selectors, and later rules.

So

```
h1 {font-weight: bold}
h1 {font-size: 200%}
```

is equivalent to

```
h1 {font-weight: bold; font-size: 200%}
```

30: Styling text

There are a number of properties for affecting the style of text:

- font-size, font-weight, font-style, font-family, and font-variant
- line-height, vertical-align, word-spacing, letter-spacing
- text-align, text-decoration, text-transform, text-indent

31: font-size

- You can use absolute or relative sizes. Relative sizes are in relation to the parent element (e.g. <body>)
- Example: `h1 {font-size: 200%}`
- Example absolute sizes:

```
10pt, small, medium, large,  
x-small, xx-small, x-large, xx-large
```

- Example relative sizes:

```
larger, smaller, 120%, 1.2em
```

- Initial value is *medium*

32: Warning about 'initial values'

- Most CSS properties have 'initial values'
- 'Initial value' means 'if no other value has been assigned'
- For HTML (but not XML) most values have been assigned by the browser already
- Example: 'font-size' has an initial value of 'medium', but the browser will likely have set a larger value for <h1>

33: Lengths

Relative:

- Ems: 4em
- X height: 1ex
- Percentages: 120%

Absolute:

- Pixels: 12px
- Inches: 0.5in
- Cm: 2.5cm, Mm: 25mm
- Points: 10pt (72pt = 1in)
- Picas: 2pc (1pc = 12pt)

The only length you may use without any units is 0.

34: Warning about font sizes

Don't use pixels for font sizes, because you don't know the resolution of the screen that anyone is using.

For instance, `font-size: 12px` will give you

- 12 point on a 72dpi screen
- 9 point on a 96dpi screen
- 6.9 point on a 125dpi screen

Similarly, avoid using point sizes as well:

- Many machines are not properly set up (the operating system doesn't know the screen resolution)
- Many people can't see text below a certain size, and so increase the base font-size

Respect these people: try to use relative font sizes. Use `font-size: 100%` for paragraph text, `font-size: 120%` or so for headings, `font-size: 80%` for copyright statements, etc.

35: font-weight

- Values: normal, bold, bolder, lighter, 100, 200, ..., 900
- normal = 400
- bold = 700
- Initial is *normal*
- Example: h1, h2, h3 {font-weight: bold}

36: font-style

- Values: *normal*, *italic*, *oblique*
- Initial: *normal*
- If you specify *italic*, but the font only has an oblique, you get that (but not vice versa)
- Example: `em {font-style: italic}`

37: font-family

- Values: a list of font names, followed by a *generic font*
- Generic fonts are: *serif*, *sans-serif*, *monospace*, *cursive*, *fantasy*
Serif, *sans-serif*, *monospace*, *cursive*, *fantasy*
- Each font in the list is tried in turn until one is found
- Example:

```
h1, h2, h3 {font-family: arial, helvetica,  
                sans-serif}
```

- Initial value depends on browser

You should always end with a generic family

38: Colours: *color* and *background-color*

- The foreground colour (text, borders, etc) is given with the *color* property
- The background colour is given with the *background-color* property
- Values are 16 colour names: black, white, gray, silver, red, maroon, yellow, olive, green, lime, blue, navy, purple, aqua, fuchsia, teal, or #F00, #FF0000, rgb(255, 0, 0), rgb(100%, 0, 0)
- Example:

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

39: Practical 1

Make a stylesheet that gives `` elements a yellow background colour, and makes `` elements be white text on a black background.

40: Class Selectors

- If an element has a *class* attribute, you can select on it
- In the CSS:

```
p.important { color: red }
```

- In the HTML:

```
<p class="important">Do not phone  
before 09:00!</p>
```

- or all "important" elements regardless of type:

```
.important { color: red }
```

Does not work for XML

41: Use of HTML: span

- Use the `` element as a carrier of *class* information:
Do `not` cross
- If you want such text to be styled in some way on non-CSS browsers as well, use `` or `` instead:
Do `<em class="important">not` cross
Do `<strong class="important">not` cross
- An element can have several classes:
`<p class="note important"> ...`

Advice: use class names that describe the purpose, not the presentation. For instance, use `class="important"`, not `class="red"`.

42: ID Selectors

- You can select any element with a particular ID tag with #:

```
#dateline { .... }  
  
<p id="dateline">Monday 12th May ...
```

- or a particular type of element with an ID:

```
h2#index { .... }  
  
<h2 id="index">Index</h2>
```

May work for XML, but no guarantee

Use an ID selector to document that there can only be one in the document, while there can be any number of elements using the same class.

43: Contextual Selectors

These allow you to address the nesting of the document:

```
h1 { font-weight: bold }  
em { font-style: italic }  
  
<h1><em>Now</em> is the time!</h1>
```

Now is the time

```
h1 em { font-weight: normal }
```

Now is the time

44: Examples of contextual selectors

- `em { font-style: italic }`
- `em em { font-style: normal }`
Nested em's revert to normal font
- `ul li { font-size: medium; color: red }`
- `ul ul li { font-size: small }`
- `ul ul ul li {font-size: x-small }`
Nested unordered lists use smaller fonts
- More specific selectors take precedence (more later), but all selectors apply. So since the first `ul li` rule sets the colour to red, all nested li's are red too.

45: Inheritance

- Note that in the following the `` element is also blue. It is *inherited* by the `` element.

```
h1 { color: blue }  
<h1><em>Now</em> is the time</h1>
```

Now is the time

- Many properties are inherited, but some are not, such as borders:

Now is the time

If the border-style were inherited by the ``, you would get:

Now is the time

46: Use of HTML: div

- Like `` for inline text, use `<div>` to carry class or ID information for larger blocks:

```
<div class="chapter">
  <h2>Chapter 2</h2>
  <p>It was dark. ... </p>
  ...
</div>

div.chapter h2 {font-family: pembo, cursive}
```

47: display

- Some elements (like ``, ``) are inline. Others (like `<p>`, `<h1>`) are blocks. The *display* property specifies this for the presentation
- Values: block, inline, list-item, none
- **block**: says that the element represents a block
- **inline**: that the element represents inline text
- **list-item**: that the element is a list item (`` in HTML) (more properties later)
- **none**: the element is not displayed at all.
- Initial value: not important for HTML; different for CSS1 and CSS2, so **never assume a default!**

48: Example of display: none

In the CSS:

```
.notcss {display: none}
```

In the HTML:

```
<p class="notcss">  
    Your browser doesn't support CSS  
</p>
```

49: Examples of display: none

You can use `display: none` with different stylesheets, to make different parts of the document visible. E.g. one stylesheet can show deletions with the text ~~struck through~~:

```
del { text-decoration: line-through }
```

another can make the deletions not visible:

```
del { display: none }
```

And we'll be using it later for tricks like this: [Hover here]

NB with:

```
body {display: none}  
h1 {display: block}
```

the h1's are still invisible, since the whole body is invisible

50: text-align

- Values: justify, left, right, center
- Applies to blocks (i.e. elements with display: block or list-item)
- Initial: not defined

At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit.

At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit.

At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit.

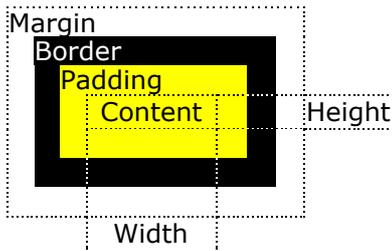
At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit.

51: Box model



- All elements have this box model
- The margin's colour is transparent
- The border's colour is by default the same as the text of the content, but can be set separately
- The padding has the same colour as the background colour of the content.

52: height and width properties



- The 'height' and 'width' properties of an element affect the height and width of the 'content' part
- So if you set width to 4em, padding to 3em, border to 2em, and margin to 1em, the whole box will be $(4 + 2 \times 3 + 2 \times 2 + 2 \times 1) = 16\text{em}$ wide. (But see note later on 'auto' values).

Note major bug in early versions of IE, where width affected the whole box. This still shows up in modern IE if you forget the DOCTYPE.

53: Box model: margin, border, padding

- All three are changeable:

```
h1 { margin: 2pt;
      border-width: 4pt;
      border-style: dotted;
      padding: 3pt}
```

- Also per part:

```
h1 { border-top: 4pt solid red}
```

also -left, -right and -bottom

54: Margins: margin-top, -right, -bottom, -left

- Examples of values: 0, auto, 2em, 3pt, 1%, ...
- Initial: 0
- Margins are in relation to enclosing element
- Percentage values refer to width of containing element
- Negative margins are allowed!
- Margins are transparent, so the enclosing element's background shows through
- *auto* means 'as calculated by the browser' (see *width*).

Example:

```
p { margin-left: 3em }
```

55: Warnings about use of margins

1.

```
body {margin-left: 4em}
h1 {margin-left: -4em}
```

<h1> typically has a larger font-size to <body>, therefore the '-4em' on h1 is **larger** than the 4em on <body>

2.

```
body {margin-left: 4em}
h1 {margin-left: 0}
```

h1 will have the **same** indent as the body (margins are relative to the parent element, not the screen)

56: Shortcuts: margin

- There are a number of shortcuts for some properties. For margins you can set all 4 sides at once:

```
margin: 1em (sets all 4 to 1em)
margin: 0 1em 0 2em
```

- The four values go clockwise and set *top right bottom left* respectively (TRBL: mnemonics treble, tribal, terrible, true-blue)
- Missing values are obtained from the opposite side: "margin: 0 1em" is the same as "margin: 0 1em 0 1em"

57: Use of margins

- Use margins for indenting:

```
    Lorem ipsum dolor sit amet, consetetur
    sadipscing elitr, sed diam nonumy
    eirmod.
```

```
    At vero eos et accusam et justo duo
    dolores et ea rebum. Stet clita kasd
    gubergren, no sea takimata sanctus est
    Lorem ipsum dolor sit.
```

```
    Invidunt ut labore et dolore magna
    aliquyam erat, sed diam voluptua.
```

- extending (using negative margins):

```
    Lorem ipsum dolor sit amet, consetetur
    sadipscing elitr, sed diam nonumy
    eirmod.
```

```
    At vero eos et accusam et justo duo
    dolores et ea rebum. Stet clita kasd
    gubergren, no sea takimata sanctus est
    Lorem ipsum dolor sit.
```

```
    Invidunt ut labore et dolore magna
    aliquyam erat, sed diam voluptua.
```

- adding space between paragraphs
- etc.

58: When vertical margins meet

When two margins meet vertically, only the larger is used (so the gap between a heading and the following paragraph is the larger of the heading's *margin-bottom* and the paragraph's *margin-top*)

An Example

When two margins meet vertically, only the larger is used (so the gap between a heading and the following paragraph is the larger of the heading's *margin-bottom* and the paragraph's *margin-top*)

An Example

When two margins meet vertically, only the larger is used (so the gap between a heading and the following paragraph is the larger of the heading's *margin-bottom* and the paragraph's *margin-top*)

59: Padding: padding-top, -right, -bottom, -left



- These properties are similar to margins
- Examples of values: 0, 2em, 3pt, 1%, ...
- Initial: 0
- Percentages refer to parent element's width

60: Padding

- Negative values are *not* allowed
- Padding takes the colour of the element's background
- Property *padding* works like *margin*, and has up to 4 values (TRBL)

```
padding-top: 1em  
padding: 1em 0em 2em 1em
```

61: Example of padding

```
blockquote  
{ margin: 2em;  
  background-color: yellow;  
  padding: 2em  
}
```

Lorem ipsum dolor sit amet, consetetur
sadipscing elitr, sed diam nonumy
eirmod tempor invidunt ut labore et
dolore magna aliquyam erat, sed diam
voluptua.

Lorem ipsum dolor sit
amet, consetetur
sadipscing elitr, sed diam
nonumy eirmod tempor
invidunt ut labore.

At vero eos et accusam et justo duo
dolores et ea rebum. Sit clita kasd
gubergren, no sea takimata sanctus est
Lorem ipsum dolor sit amet.

62: Borders: width, style and color



Borders can have a *width*, *style* and *color*. For widths:

- Properties: `border-top-width`, `-bottom-width`, `-right-width`, `-left-width`
- Example values: `thin`, `medium`, `thick`, `1pt`, `1em`, ...
- Initial: `medium` (but see *border-style*)
- Example:

```
border-left-width: 1pt
```

63: Shorthand: border-width

- Property *border-width* can have up to 4 values, just like *margin* and *padding* (TRBL)
- Example:

```
border-width: 1pt 2pt
```

- So top, bottom=1pt,
- right, left= 2pt

64: border-style

- Property: border-style
- Values: none, dotted, dashed, solid, double, groove, ridge, inset, outset
- Initial: none
- Sets value for all 4 sides! (But see *border-top*, *border-right*, *border-bottom*, *border-left*)



65: border-color

- Property: border-color
- Values: one to four colours (see *color* property)
- Initial: whatever value the *color* property has for this element
- Four values work like *margin* (etc): TRBL
- Example:



(left side is thus also white)

66: Shorthands: border-top, -right, -bottom, -left

- Values: *width style colour*
- Example:

```
p.note {border-left: medium solid black}
```

- Initial: as individual properties
- Values may be in any order (border-top: thin red 1pt)
- Any of the three values may be left out (but see warning later):

```
border-top: thin blue
```

67: One last border shorthand: border

- Property: border
- Values: *width style color*
- Values may be in any order, and any may be omitted (but see warning)
- Sets all 4 sides
- Example

```
p.warning {border: solid thick red}
```

68: Warning: border-style

If you set `border-width`, or `border-color`, and forget to set `border-style`, since the default is `'border-style: none'` **you will see no border!**

Always set `border-style` if you want a border.

69: Warning about using shorthands

- `border` and `border-top` (etc) also set the colour, so with:

```
blockquote {color: black;
            border: red medium solid;
            border-left: dotted }
```

even though the colour isn't mentioned in the `border-left` property, it is there! And its value is the value of `color`: therefore the left border will be black.

Better to be explicit.

70: Usage of borders

Use borders for:

| Setting off text with a line each side |

Enclosing text in a box

Putting a line under a paragraph

| Marking changed paragraphs with a line

A border will often be too close to the text: use padding to set it off from the text:

The End

The End

71: height and width

The height and width of elements is normally determined by context or by the element itself.

For instance, for text, the width is determined by the width of the window, and the height by the amount of text.

Images have an inbuilt size.

You can change these defaults with the height and width properties.

- Property: height
- Values: auto, 100px, 15em, ... (no percentages)
- Initial: auto

72: width

- Property: width
- Values: auto, 100px, 15em, 50%, ...
- Initial: auto
- Percentages: refer to parent's width
- auto: calculated size, or intrinsic width for images.
- Example, to create a page of thumbnails:

```
img { width: 25% }
```

height is *auto* so will also scale to preserve aspect ratio

73: Auto values for box model



- Normally 'width' is 'auto'
- If no value is 'auto', margin-right will be set to 'auto'

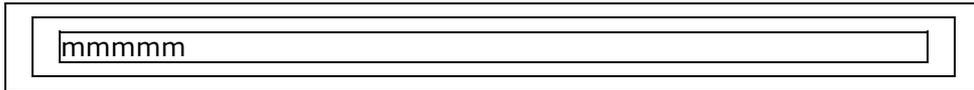
74: Example

```
<div class="outer">  
  <div class="middle">  
    <div class="inner">mmmmm</div>  
  </div>  
</div>
```

with

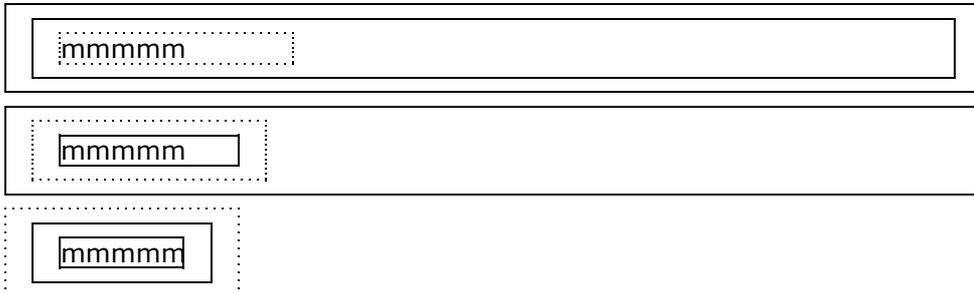
```
div {margin: 0.5em 1em;  
     border: thin black solid}
```

gives:



75: Example (contd)

Setting respectively inner, middle and outer to width: 9em gives:



76: Practical 2

Make a stylesheet that indents all text except headings.

Make a stylesheet that hides all text except headings. Indent headings according to their importance (don't indent <h1>; indent <h2> a little; <h3> a little more, etc.)

77: Text properties: line-height

- The line-height is the distance between the base of one line, and the base of the next.
- Example values: normal, 1.2, 120%, 1.2em, 12pt, ...
- Initial: normal (browser specific)
- **Better to use relative values**
- If font-size is 10pt, then a line-height specified as 1.2, 120% or 1.2em would result in a line-height of 12pt. The extra space is equally spread above and below the line. (This paragraph has line-height: 100%, so should look a bit compressed)

78: Warning about line-height

- There is a difference in inheritance: a number (e.g. 1.2) is inherited by the children, but in the case of other factors (120%, 12em), the *resulting value* (e.g. 12pt) is inherited. If the child has a different font-size, but no specified line-height, it may look wrong. **If in doubt, use numbers.**

```
body {font-size: 10pt; line-height: 1.2}
h1 {font-size: 20pt}
```

h1 has a line-height of $20\text{pt} \times 1.2 = 24\text{pt}$

```
body {font-size: 10pt; line-height: 1.2em}
h1 {font-size: 20pt}
```

h1 inherits the same line-height as body, which is $10\text{pt} \times 1.2\text{em} = 12\text{pt}$

79: text-decoration

- Values: *none*, or any combination of: *underline*, *overline*, *line-through*, *blink*
- Initial: *none*
- Not all browsers implement blink.
- Example:

```
a {text-decoration: underline}
```

underline, overline, ~~line-through~~, mixture

80: text-indent

- This specifies the indentation of the first line of a block of text
- Example values: 0, 4em, 1%, ...
- Initial: 0
- Use negative values for extending a line.

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

81: word-spacing, letter-spacing

- These are used to stretch a number of words or compress a number of words by adding extra spacing between letters or words
- Values: normal, 1px, 0.1em, ...
- Not widely implemented

82: vertical-align

- For effects like subscript and superscript
- Values: baseline, sub, super, top, text-top, middle, bottom, text-bottom
<percentage> (of line height; this is 100%)

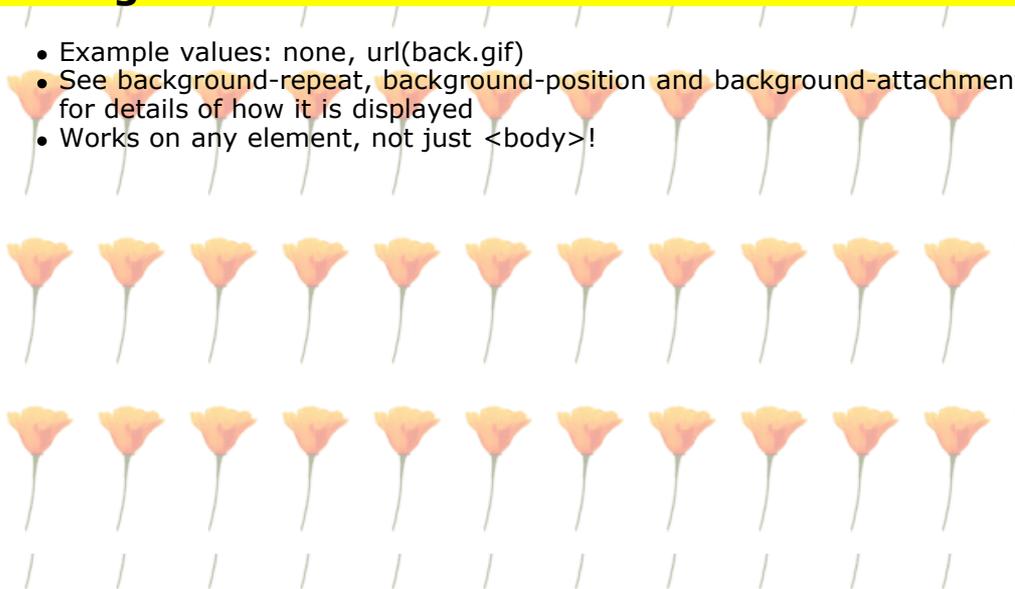
Values:

- Initial: baseline
- Only trust baseline, sub, and super at present
- Example:

```
sub {vertical-align: sub}
```

83: Background properties: background-image

- Example values: none, url(back.gif)
- See background-repeat, background-position and background-attachment for details of how it is displayed
- Works on any element, not just <body>!



84: background-position

- Specifies where background image is to be placed, or where repeating is to start from
- Example values: 0% 0%, top left, center, any reasonable mixture of top, bottom, center, left, right, ...
- Initial: 0% 0% (=top left)
- This example is bottom right



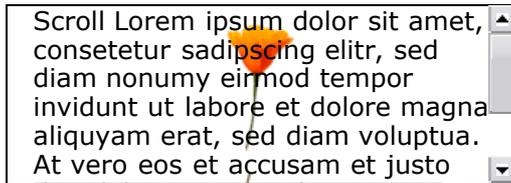
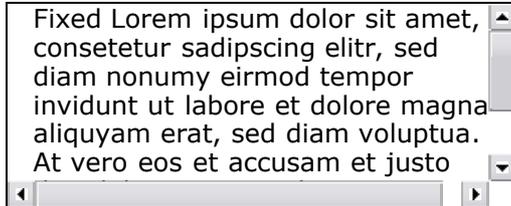
85: background-repeat

- Specifies how background image is to be displayed
- Values: repeat, no-repeat, repeat-x, repeat-y
- no-repeat: just once at start position
- repeat-x: repeat horizontally both sides of the start position
- repeat-y: repeat vertically above and below start position
- repeat: repeat in all directions (tile the element)
- Initial: repeat



86: background-attachment

- Specifies if the background scrolls with the page, or stays put
- Values: scroll, fixed
- Initial: scroll
- Use for instance to put a logo or water-mark that remains visible when scrolling



87: Technique: Images for headlines

All too often Websites use images for major headings. This causes some problems:

- Google doesn't see the text, nor know it's a heading
- Other software doesn't know it's a heading
- Causes problems with accessible software
- Doesn't scale on the screen
- Doesn't work with browsers that don't handle images.

However, there is a technique that gives you the best of both worlds: put the text in the HTML and override it with an image in the CSS

88: *images for Headlines*

In the HTML, use a heading, with the text in a span:

```
<h1><span>Images for headlines</span></h1>
```

In the CSS, turn off the span:

```
h1 span {display: none}
```

and add a background image to the h1:

```
h1 { background-image: url(text.gif);  
      background-repeat: no-repeat;  
      background-position: 50% 50%  
      height: 80px; width: 80px; }
```

89: Pseudo Classes: Anchors

```
a:link { color: blue }  
a:visited { color: #f0f }  
a:active { color: red }  
a:link img { border-style: solid;  
             border-color: blue }
```

- CSS2, but useful: when the mouse is over an element:

```
a:hover {background-color: yellow}
```

90: Note on <a>

```
a {color: green}
a:link {color: blue}
```

This will colour elements green, and elements blue.

Beware!

```
p {color: red}
a:link {color: blue}
```

```
<p><a href="...">Click here</a></p>
```

"Click here" will be blue.

91: Pseudo element: first-line, first-letter

- **N**OTE THERE ARE ALSO SELECTORS TO SELECT THE FIRST LINE and first letter of the *formatted* output:

```
p:first-line {font-variant: small-caps;
              color: blue}
p:first-letter {font-size: 200%;
               color: red}
```

- Not widely implemented

92: font-variant

- Values: normal, small-caps
- USES A SMALL-CAPS VARIANT OF THE FONT
- Initial: normal

93: Float

- Move elements relative to the parent
- Values: none, left, right
- Example. Logo always on the side of the window:

```
img.logo {float: right}
img.logo {float: left}
```

- Initial: none

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, no sea takimata sanctus est Lorem ipsum dolor sit amet. *** Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit.



Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, no sea takimata sanctus est Lorem ipsum dolor sit amet. *** Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit.



94: Use of float

- Float works on any type of element, even text blocks!
- It can replace the use of tables for some layout

```
p.menu {float: left;
        background-color: yellow;
        ...
}
```

```
<p class="menu">Menu items ...</p>
<p>Text ...</p>
```

```
line 1 Lorem ipsum dolor sit amet,
line 2 consetetur sadipscing elitr,
line 3 sed diam nonumy eirmod
line 4 tempor invidunt ut labore et
line 5 dolore magna aliquyam erat,
sed diam voluptua. At vero
eos et accusam et justo duo
dolores et ea rebum. Stet citta kasd
gubergren, no sea takimata sanctus est
Lorem ipsum dolor sit amet.
```

95: Technique: multi-columns using float

```
div.col {
  width: 31%;
  text-align: justify;
  padding-left: 1%;
  padding-right: 1%;
  border-left: solid 1px black;
  margin: 1em 0;
  float: left;
}
div.col.first {border-style: none}
```

96: An example of columns

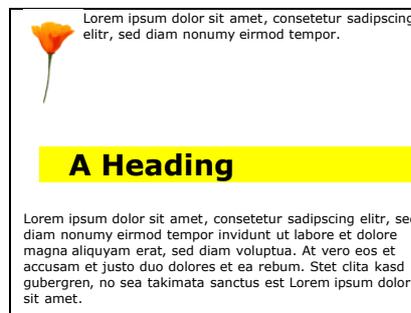
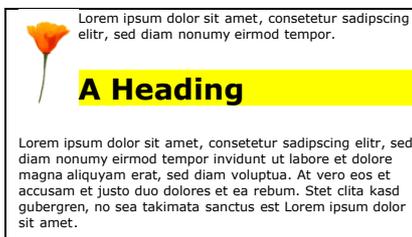
Here is an example of columns using CSS:

This is a div of class = "first col". Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna.	This is a div of class = "col". Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna.	This is a div of class = "col". Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna.
---	---	---

And then back to single column format after the divs.

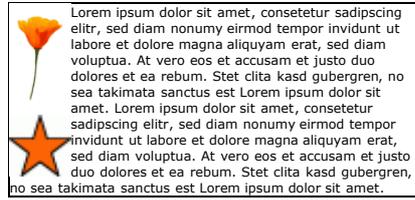
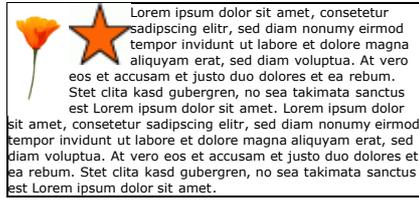
97: clear

- Allows an element to refuse floating elements one side or another
- Values: none, left, right, both
- Initial: none
- Example: h2 {clear: left}



98: Use of clear

Both of the images are at the start of the paragraph (in the source). The flower is first, then the star; the left example has `img {float: left}` and the right has `img {float: left; clear: left}`:



99: text-transform

- Values: none, uppercase, lowercase, capitalize
- Example: h1, h2, h3 {text-transform: uppercase}
- Initial: none
- Examples: none, UPPERCASE, lowercase, Capitalize

100: font

- This is a shorthand for font-style, font-variant, font-weight, font-size, line-height, font-family
- Example:

```
p.abstract
  {font: medium italic 10pt/12pt
    times, serif }
```

101: white-space

- Values: normal, pre, nowrap
- pre: use for <pre> like elements
- nowrap: text doesn't get wrapped

Remember: there is nothing inherent in the <pre> element that causes it to retain layout and output in a monospaced font: it is the styling that does that. You can change it.

102: list-style-type

- Values: disc, circle, square, decimal, lower-roman, upper-roman, lower-alpha, upper-alpha, none
- Applies to elements with display: list-item
- Example:

```
ul li {list-style-type: disc}
ul ul li {list-style-type: circle}
ol li {list-style-type: decimal}
```

103: list-style-type: examples

This is a single with different list-style-types applied to each :

- disc
- circle
- square
- none
- 5. decimal
- VI. upper-roman
- vii. lower-roman
- H. upper-alpha
- i. lower-alpha

104: list-style-image

- ★ Example values: none, url(star.gif)
- ★ Allows you to define your own sort of bullets
- ★ Example:

```
ul.special {  
    list-style-image: url(ball.gif)}
```

105: list-style-position

- Values: inside, outside
- Default: outside
- Defines whether the bullet goes inside or outside the text box. As you may be able to see, this line has a value for list-style-position of 'outside'.
- And if you compare this bulleted item with the one above, you should be able to see that it has a value for list-style-position of 'inside'.

106: list-style

This is a shorthand

- Values: *type style <url> position*
- Example:

```
ul.special {  
    list-style: url(ball.gif) inside}
```

107: Cascading

- A browser has an implicit default style sheet
- The user may have a preferences style sheet
- Browser default → Reader preferences → Document
- You can combine several style sheets (= cascading)

```
@import url(house.css);  
body { font-size: 30pt }
```

- You can override with !important:

```
body {color: black !important;  
    background-color: white !important}
```

108: Selectivity of selectors

- !important wins
- browser → user → document
- id > class > no of tags in contextual selector
- pseudo-element = normal element, pseudo-class = normal class
- Last specified wins if otherwise equal
- CSS rules win over HTML attributes (like bgcolor, align)

109: Optional Stylesheets

```
<link rel="stylesheet"
      type="text/css" href="main.css">
<link rel="alternate stylesheet" title="Big"
      type="text/css" href="alt.css">
<link rel="stylesheet" media="print"
      type="text/css" href="print.css">
```

Media: all, screen, print, projection, handheld, tv, tty, aural, braille, embossed.

HTML defines the default as "screen"

You can also use it on the style element:

```
<style type="text/css" media="print">
  ...
</style>
```

110: Practical 3

Here is a [document](#) styled with HTML and images.

Do some of the obvious parts with CSS (as much as you have time for).

111: Implementation

Already available in:

- Microsoft IE 3, 4, 5, 6
- Netscape 4, 6, 7, 8
- Opera 3, 4, 5, 6, 7, 8
- X-Smiles
- NetClue
- OmniWeb
- Mozilla, Galeon, Firebird, K-Meleon, Chimera, ...
- Konqueror
- Safari

112: Implementation

- Escape
- Icab
- Millions of mobile phones
- Arachne
- Emacs-w3
- Amaya
- Athena
- Closure
- HP ChaiFarer
- ICE

113: Level Compatibility

- All CSS1 rules are acceptable to CSS2 processors
- If a CSS1 browser comes across a CSS2 selector, it ignores the whole rule
- If a CSS1 processor comes across a CSS2 property or value, it ignores only the declaration.

Ignore rule:

*[width] {font-size: 10pt; color: blue}

Ignore declaration:

p {overflow: hidden; color: blue}

Ignore declaration:

h2 {display: run-in; color: blue}

114: CSS2 and 3

Later areas of work include:

- Speech
- Layout
- Fuller control
- Printing
- ...

CSS2: Selectors

More selectors, such as:

- *: any element
- E > F: selects any element F that is a child of E
- E:first-child: selects any E that is the first child of its parent (To select the first child of an element E use "E > :first-child")
- E + F: any F that directly follows an E
- E[att]: any E that has attribute att
- E[att="value"]: where att has exactly the value
- E[att~="value"]: where one of the words of att is the value

CSS2: Features

- Embedded media rules: all, screen, print, projection, handheld, tv, tty, aural, braille, embossed

```
@media print { body {font-size: 10pt} }  
@media screen { body {font-size: 12pt} }  
@media projection { body {font-size: 20pt} }  
h1 {font-size: 2em}
```

- *inherit* as value for all properties
- Control of tables
- Page layout
- Bi-directional text
- Web fonts
- Aural style sheets: `h1 { voice-family: male; pitch: low }`

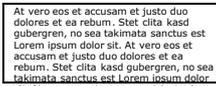
CSS2: More features

- Generated content, and control over counters
- Text shadows
- Borders and padding properties fixed up; outlines
- `display`: new values
- Absolute and relative positioning of elements; z-index for overlapping
- Overflow control and clipping
- Font stretching and adjusting
- System colours, more units, more to font, list-style-types
- + details

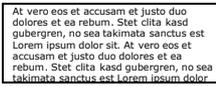
CSS2: Overflow

Values: visible, hidden, scroll, auto, inherit
Default: visible

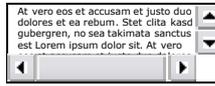
- Visible: show even if it doesn't fit
- Hidden: hide what doesn't fit
- Scroll: use scroll bars regardless
- Auto: use scrollbars if needed

A rectangular text box containing multiple lines of Lorem Ipsum text. The text is visible and wraps within the box boundaries, demonstrating the 'visible' overflow behavior.

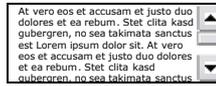
At vero eos et accusam et justo duo dolores et ea rebum. Stet citta kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit. At vero eos et accusam et justo duo dolores et ea rebum. Stet citta kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit.

A rectangular text box containing multiple lines of Lorem Ipsum text. Only the first line is visible, and the rest is hidden, demonstrating the 'hidden' overflow behavior.

At vero eos et accusam et justo duo dolores et ea rebum. Stet citta kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit. At vero eos et accusam et justo duo dolores et ea rebum. Stet citta kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit.

A rectangular text box containing multiple lines of Lorem Ipsum text. The text is visible and wrapped, with a vertical scrollbar on the right side, demonstrating the 'scroll' overflow behavior.

At vero eos et accusam et justo duo dolores et ea rebum. Stet citta kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit. At vero eos et accusam et justo duo dolores et ea rebum. Stet citta kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit.

A rectangular text box containing multiple lines of Lorem Ipsum text. The text is visible and wrapped, with a vertical scrollbar on the right side that only appears when needed, demonstrating the 'auto' overflow behavior.

At vero eos et accusam et justo duo dolores et ea rebum. Stet citta kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit. At vero eos et accusam et justo duo dolores et ea rebum. Stet citta kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit.

CSS2: Position

Values: static, relative, absolute, fixed, inherit
Default: static

- Static: not positioned
- Relative: positioned relative to current position (like sub/superscripting)
- Absolute: positioned absolutely in the document (relative to closest enclosing absolutely positioned box, or otherwise the whole document)
- Fixed: positioned absolutely on the screen

Uses properties:

- top, bottom, left, right: the offset of the positioned box from the position is is being positioned against.

CSS2: Technique – a fixed list of links

See [example](#). HTML:

```
<div class="nav">
  <h2>Links</h2>
  <p>
    <a href="#">Home</a>
    <a href="#">Away</a>
    ...
  </p>
</div>
```

And then in the CSS:

```
.nav { position: fixed;
       top: 0; right: 0;
       width: 6em;
       ...}
.nav a {display: block}
```

121: Technique: Hover pop-ups

Example

This contains markup like

```
<a class="popup" href="#">Example<span>Popup text
here</span></a>
```

and rules like

```
a.popup span {display: none}
a.popup:hover span {
  display: block; width: 10em;
  background-color: yellow; border: dotted black thin;
  position: absolute; left: 40%;}
```

CSS2: Generated content

You can generate content that isn't in the document.

For instance, this slide does not have the initial "CSS2: " in the header, but a rule that says

```
h2.CSS2:before {content: "CSS2: "}
```

There is also a ":after".

CSS2: Numbering

In the printed version of these slides, each slide has been given a number.

This is done using rules like:

```
div.slide h2 {counter-increment: slide}  
div.slide h2:before{ content: counter(slide) " "}
```

124: **CSS 2.1**

Coming soon: a cleaned up version of CSS2

- A few new property values
- A few changes
- A few deletions

125: **CSS3**

- Printing
- Multicolumn
- Headers and footers
- More media queries
- ...

126: Where?

The definition of CSS1 can be found at: <http://www.w3.org/TR/REC-CSS1>

The definition of CSS2 is at: <http://www.w3.org/TR/REC-CSS2/>

The definition of CSS2.1 is at: <http://www.w3.org/TR/CSS21/>

See also: <http://www.w3.org/TR/CSS21/changes.html>

CSS resources can be found at www.w3.org/Style/CSS

127: Future Markup

- HTML was designed for just one sort of document (scientific reports), but is now being used for all sorts of different documents
- You could use SGML to define other sorts of document, but SGML is notoriously hard to fully implement

128: XML

- XML is a W3C effort to simplify SGML
- It is a meta-language, a subset of SGML
- One of the aims is to allow everyone to invent their own tags
- DTD is optional: a DTD can be inferred from a document

129: Consequences

- The requirement of being able to infer a DTD from a document has an effect on the languages you can define:
- Closing tags are now required
.... <P>....</P>
- Empty tags are marked specially

 <HR/> (or <HR></HR> etc)

130: Consequences 2

- CDATA sections must be marked as such (if they contain "<", "&" etc.):

```
<SCRIPT>
<![CDATA[
    ... script content ...
]]>
</SCRIPT>
```

131: Consequence of XML

- Anyone can now design a (Web-delivered) language
- CSS makes it viewable

```
<address>
<name>Steven Pemberton</name>
<company>CWI</company>
<street>Kruislaan 413</street>
<postcode>1098 SJ</postcode>
<city>Amsterdam</city>
<speaker/>
</address>
```

132: So do we still need HTML?

- XML is still a meta-language
- There is still a perceived need for a base-line mark-up
- HTML has some useful semantics, both implied and explicit (search engines gladly use it, for instance)

133: HTML as XML application

- Clean up (get rid of historical flotsam)
- Modularise – split into separate parts
- Allows other XML applications to use parts
- Allows special purpose devices to use subset
- Add any required new functionality (forms, better event handling, Ruby)

134: Differences HTML:XHTML

- Because of the difference between SGML and XML, there are some necessary differences, for instance:
- Use lower case: <p> not <P>
- Attributes are always quoted:
<th colspan="2">
- Anchors use *id* attribute not *name* (and not just on <a> by the way):
 <p id="top">

135: Examples

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Virtual Library</title>
  </head>
  <body>
    <p>Moved to <a href="http://vlib.org/">vlib.org</a>.
  </p>
</body>
</html>
```

136: Namespaces

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>A Math Example</title></head>
<body>
  <p>The following is MathML markup:</p>
  <math xmlns="http://www.w3.org/TR/REC-MathML">
    <apply><log/><logbase><cn> 3 </cn> </logbase>
      <ci> x </ci>
    </apply>
  </math>
</body>
</html>
```

137: Transition

- XHTML 1.0 has been carefully designed to make use of 'quirks' in existing HTML browsers
- Use of a small number of guidelines allows XHTML to be served to HTML browsers

138: Examples of Guidelines

- Use space before / of empty elements:

```
<br /> <hr /> 
```

- Use *name=* and *id=* on <a>:

```
<a name="index" id="index"> ... </a>
```

139: Versions of XHTML

There are now several versions of XHTML in use:

- XHTML 1.0: 'Legacy-compatible' version.
- XHTML 1.1: Cleaned up version of XHTML 1.0 Strict
- XHTML Basic: For small devices. Now in many [WAP 2 phones](#).

There are also versions for TVs and printers in preparation, as well as XHTML 2.0, with many new features.

140: Result

- XML with related technologies gives you the freedom to define and deliver your own document types
- HTML is still needed as a base-line markup
- The new HTML gives a transition path to the future
- Since there is no built-in presentation semantics any more, CSS is *essential*

141: Reminder: Using style sheets with XML

For XML use a processing instruction:

```
<?xml-stylesheet type="text/css"
  href="your-filename.css"?>
```

Put before first element of the document

142: Practical 4: RSS

An **RSS document** is an XML document that contains elements like this:

```
<item rdf:about="http://www.w3.org/News/2004#item155">
  <title>W3C Celebrates Ten Years Leading the Web</title>
  <description>This year, W3C celebrates its tenth anniversary. ...
</description>
  <link>http://www.w3.org/News/2004#item155</link>
  <dc:date>2004-10-07</dc:date>
</item>
```

Write a stylesheet to make such a document readable.

143: Overview of properties, with examples and defaults

- font-*:
 - family (*Futura, ..., serif, sans-serif, cursive, fantasy, monospace*)
 - style (**normal**, *italic, oblique*)
 - variant (**normal**, *small-caps*)
 - weight (**normal**, *bold, bolder, lighter, 100, ..., 400, ..., 900*)
 - size (*10pt, 120%, small, medium, large, smaller, larger, ...*)
- color (*red, ..., #f00, #ff0000, rgb(255,0,0), rgb(100%, 0, 0), ...*)
- background-*:
 - color (**transparent**, *red, black, white, gray, silver, red, maroon, yellow, olive, green, lime, blue, navy, purple, aqua, fuchsia, teal, ...*)
 - image (*none, url(back.gif)*)
 - repeat (**repeat**, *no-repeat, repeat-x, repeat-y*)
 - attachment (**scroll**, *fixed*)
 - position (**0% 0%**, *top left, center, center left, bottom right, ...*)
- line-height (**normal**, *120%, ...*)
- word-spacing, letter-spacing (*normal, 1%, 1px, ...*)
- vertical-align (**baseline**, *sub, super, 10%, top, text-top, middle, ...*)
- text-*:
 - decoration (**none**, *underline, overline, line-through, blink*)
 - transform (**none**, *uppercase, lowercase, capitalise*)
 - align (*justify, left, right, center*)
 - indent (**0**, *4em, ...*)
- display (**block**, *inline, list-item, none*)
- white-space (**normal**, *pre, nowrap*)
- list-style*:
 - type (**disc**, *circle, square, decimal, none, lower-roman, lower-alpha, ...*)
 - image (*url(sphere.gif)*, **none**)
 - position (*inside*, **outside**)
 - list-style (*type position <url>*)

144: Overview of box properties

- margin-*: top, right, bottom, left (**0**, auto, 2em, 3pt, 1%, ...)
- padding-*: top, right, bottom, left (**0**, 2em, 3pt, 1%, ...)
- border-*:
 - width⁴ (thin, **medium**, thick, 2pt, ...)
 - style (**none**, dotted, dashed, solid, double, ...)
 - color⁴ (...)
 - top, right, bottom, left (*width style colour*)
 - top-width, bottom-width, right-width, left-width (**medium**, ...)
- margin⁴ (*top right bottom left*)
- padding⁴ (*top right bottom left*)
- border (*width style color*)
- height, width (**auto**, 100px, 15em, 50%, ...)
- float (**none**, left, right)
- clear (**none**, left, right, both)

145: Web Resources for CSS, XML and XHTML

- A list of known books about CSS, online resources, supporting browsers, and editors: <http://www.w3.org/Style/CSS>.
- The CSS1 Recommendation: <http://www.w3.org/TR/REC-CSS1>
- The CSS2 Recommendation: <http://www.w3.org/TR/REC-CSS2/>
- A CSS1 Quick reference: <http://www.cwi.nl/~steven/www/css1-qr.html>
- XHTML: <http://www.w3.org/TR/xhtml1/> with more information at <http://www.w3.org/Markup>
- XML: <http://www.w3.org/TR/REC-xml> with more information at <http://www.w3.org/XML/>
- An excellent book on CSS2, is by two of its creators, Håkon Lie and Bert Bos. It is *Cascading Style Sheets: Designing for the Web*, published by Addison-Wesley. The latest edition was written entirely in XHTML and CSS.
- Validate your CSS: <http://jigsaw.w3.org/css-validator/>
- Validate your HTML and XHTML: <http://validator.w3.org/>
- Tidy up your HTML, making it more amenable for CSS, and convert it to XHTML 1.0: <http://www.w3.org/People/Raggett/tidy/>
- Test your browser for CSS compliance: <http://www.w3.org/Style/CSS/Test/>
- A set of style sheets for HTML designed by a graphic-designer: <http://www.w3.org/StyleSheets/Core/>
- How well (or badly) browsers implement CSS, feature by feature: <http://www.webreview.com/style/css1/charts/mastergrid.shtml>
- Workarounds for browser bugs: <http://css.nu/pointers/bugs.html>
- Google's directory of resources: <http://directory.google.com/Top/Computers/Programming/Internet/CSS/>.
- The 'House of Style': http://www.westciv.com/style_master/house/
- A webzine of standards-based web-building techniques: <http://www.alistapart.com/stories/>.
- Standards evangelism: <http://www.webstandards.org/>.
- Some examples of CSS-based sites: [W3C](#), [Wired](#), [webstandards.org](#), [A List Apart](#), [a Lycos redesign](#) at <http://jscript.dk/lycos/2/>, [XHTML2](#) at <http://w3future.com/weblog/gems/xhtml2.xml>.

146: Quick Reference to Cascading Style Sheets, level 1

This version is based on:

<http://www.w3.org/TR/REC-CSS1>

Latest version:

<http://www.w3.org/TR/WD-css1.html>

Author:

Steven Pemberton (Steven.Pemberton@cwi.nl)

147: Syntax

a b c

a is followed by b is followed by c, in that order.

a | b

either a or b must occur

a || b

either a or b or both must occur, in any order

[a b]

brackets, used for grouping

a?

a is optional

a*

a is repeated 0 or more times

a+

a is repeated 1 or more times

a{1,4}

a is repeated at least once and at most 4 times.

Juxtaposition is stronger than the double bar, and the double bar is stronger than the bar. Thus "a b | c || d e" is equivalent to "[a b] | [c || [d e]]".

148: Definitions

Block-level elements

an element which has a line break before and after (e.g. <H1>, <P>)

Replaced element

An element which is replaced by content pointed to from the element.
E.g., .

149: Properties

In each definition

- the default value is shown in bold, or given separately
- values apply to all elements unless otherwise stated
- properties are inherited unless the property name is marked with a star "*".

150: 5.2 Font properties

5.2.2 font-family

[[<family-name> | <generic-family>],]* [<family-name> | <generic-family>]

Initial: UA specific

<generic-family>

serif | sans-serif | cursive | fantasy | monospace

5.2.3 font-style

normal | italic | oblique

5.2.4 font-variant

normal | small-caps

5.2.5 font-weight

normal | bold | bolder | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900

5.2.6 font-size

<absolute-size> | <relative-size> | <length> | <percentage>

Initial: medium

Percentage values: relative to parent element's font size

<absolute-size>

xx-small | x-small | small | medium | large | x-large | xx-large

<relative-size>

larger | smaller

5.2.7 font

[<font-style> || <font-variant> || <font-weight>]? <font-size> [/ <line-height>]? <font-family>

Initial: not defined for shorthand properties

Percentage values: allowed on <font-size> and <line-height>

151: 5.3 Color and background

5.3.1 color

<color>

Initial: UA specific

5.3.2 background-color*

<color> | transparent

5.3.3 background-image*

<url> | none

5.3.4 background-repeat*

repeat | repeat-x | repeat-y | no-repeat

5.3.5 background-attachment*

scroll | fixed

5.3.6 background-position*

[<percentage> | <length>]{1,2} | [top | center | bottom] || [left | center | right]

Applies to: block-level and replaced elements

Initial: 0% 0%

Percentage values: refer to the size of the element itself

5.3.7 background*

<background-color> || <background-image> || <background-repeat> || <background-attachment> || <background-position>

Initial: not defined for shorthand properties

Percentage values: allowed on <background-position>

152: 5.4 Text properties

5.4.1 word-spacing

normal | <length>

5.4.2 letter-spacing

normal | <length>

5.4.3 text-decoration*

none | [underline || overline || line-through || blink]

Inherited: no, but see clarification below

5.4.4 vertical-align*

baseline | sub | super | top | text-top | middle | bottom | text-bottom | <percentage>

Applies to: inline elements

Percentage values: refer to the 'line-height' of the element itself

5.4.5 text-transform

capitalize | uppercase | lowercase | **none**

5.4.6 text-align

left | right | center | justify

Applies to: block-level elements

Initial: UA specific

5.4.7 text-indent

<length> | <percentage>

Applies to: block-level elements

Initial: 0

Percentage values: refer to parent element's width

5.4.8 line-height

normal | <number> | <length> | <percentage>

Percentage values: relative to the font size of the element itself

153: 5.5 Box properties

5.5.1 margin-top*

<length> | <percentage> | auto

Initial: 0

Percentage values: refer to parent element's width

5.5.2 margin-right*

<length> | <percentage> | auto

Initial: 0

Percentage values: refer to parent element's width

5.5.3 margin-bottom*

<length> | <percentage> | auto

Initial: 0

Percentage values: refer to parent element's width

5.5.4 margin-left*

<length> | <percentage> | auto

Initial: 0

Percentage values: refer to parent element's width

5.5.5 margin*

[<length> | <percentage> | auto]{1,4}

Initial: not defined for shorthand properties

Percentage values: refer to parent element's width

5.5.6 padding-top*

<length> | <percentage>

Initial: 0

Percentage values: refer to parent element's width

5.5.7 padding-right*

<length> | <percentage>

Initial: 0

Percentage values: refer to parent element's width

5.5.8 padding-bottom*

<length> | <percentage>

Initial: 0

Percentage values: refer to parent element's width

5.5.9 padding-left*

<length> | <percentage>

Initial: 0

Percentage values: refer to parent element's width

5.5.10 padding*

[<length> | <percentage>]{1,4}

Initial: 0

Percentage values: refer to parent element's width

5.5.11 border-top-width*

thin | **medium** | thick | <length>

5.5.12 border-right-width*

thin | **medium** | thick | <length>

5.5.13 border-bottom-width*

thin | **medium** | thick | <length>

5.5.14 border-left-width*

thin | **medium** | thick | <length>

5.5.15 border-width*

[thin | medium | thick | <length>]{1,4}

Initial: not defined for shorthand properties

5.5.16 border-color*

<color>{1,4}

Initial: the value of the 'color' property

5.5.17 border-style*

none | dotted | dashed | solid | double | groove | ridge | inset | outset

5.5.18 border-top*

<border-top-width> || <border-style> || <color>

Initial: not defined for shorthand properties

5.5.19 border-right*

<border-right-width> || <border-style> || <color>

Initial: not defined for shorthand properties

5.5.20 border-bottom*

<border-bottom-width> || <border-style> || <color>

Initial: not defined for shorthand properties

5.5.21 border-left*

<border-left-width> || <border-style> || <color>

Initial: not defined for shorthand properties

5.5.22 border*

<border-width> || <border-style> || <color>

Initial: not defined for shorthand properties

5.5.23 width*

<length> | <percentage> | **auto**

Applies to: block-level and replaced elements

Percentage values: refer to parent element's width

5.5.24 height*

<length> | **auto**

Applies to: block-level and replaced elements

5.5.25 float*

left | right | **none**

5.5.26 clear*

none | left | right | both

154: 5.6 Classification

5.6.1 display*

block | inline | list-item | none

5.6.2 white-space

normal | pre | nowrap

Applies to: block-level elements

5.6.3 list-style-type

disc | circle | square | decimal | lower-roman | upper-roman | lower-alpha | upper-alpha | none

Applies to: elements with 'display' value 'list-item'

5.6.4 list-style-image

<url> | **none**

Applies to: elements with 'display' value 'list-item'

5.6.5 list-style-position

inside | **outside**

Applies to: elements with 'display' value 'list-item'

5.6.6 list-style

<list-style-type> || <list-style-position> || <url>

Applies to: elements with 'display' value 'list-item'

Initial: not defined for shorthand properties

155: 6.1 Length units

<length>

[+|-]?<number><unit>

<number>

<digit>+[.<digit>]*?

<unit>

<absolute-unit> | <relative-unit>

<absolute-unit>

mm | cm | in | pt | pc

<relative-unit>

em | ex | px

156: 6.2 Percentage units

<percentage>

<number>%

157: 6.3 Color units

<color>

<color-name> | <rgb>

<color-name>

aqua | black | blue | fuchsia | gray | green | lime | maroon | navy |
olive | purple | red | silver | teal | white | yellow

<rgb>

#<hex><hex><hex> |

#<hex><hex><hex><hex><hex><hex> |

rgb(<number>, <number>, <number>) |

rgb(<percentage> <percentage>, <percentage>)

6.4 URL

<url>

url(<text>)