

Structure vs. Presentation

- HTML Original Intent:
 - Author: specifies document structure, not presentation
 - Browser: renders marked-up content into human-readable output
 - Formatting (layout) based on its environment: desktop, laptop, palmtop, web phone, etc.
- In Practice:
 - Designers wanted much more control over layout
 - Wanted control of colour, italics, bold, etc.
 - Pouring text into tables for alignment
 - Using 1-pixel transparent images for precise spacing and alignment
 - Browsers added growing number of ad hoc tags and attributes
 - Browser-specific extensions are a nightmare for portability
 - HTML 'degenerated' into a markup language for format
 - HTML was not designed for this purpose

1

Problems

- Reduces performance and programmer productivity
- Non standard way to add features
- Not very flexible
 - Hard to change and redesign
 - Hard to support user customizations
 - Hard to support heterogeneous devices/interfaces

2

2,400 HTML characters to describe 60 characters of content



3

Problems

- Inaccessibility
 - Hard to render on heterogeneous devices (PC, mobile phones, PDA)
 - Does not support multi-modal interfaces
 - Screen readers for visual impaired
- Limits user customization
- Reduce performance and programmer productivity
 - Redundant formatting information swells file size

4

Cascading Style Sheets (CSS)

- Separate structure from presentation
- “Simple” mechanism to attach style to structured documents
 - fonts, colours, spacing, ...
- Applies generically to all forms of XML
- Removes the requirement for further formatting tags from HTML proper

5

CSS Advantages

- Precise control over presentation
- Simplify site maintenance
- Faster downloads
- Resolution independence
- Media-specific rendering

6

CSS & XHTML

Link style sheet to XML

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

For HTML and XHTML

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

```
<style type="text/css">  
  style declarations
```

```
</style>
```

```
<h1 style="style declarations">
```

7

CSS Language

stylesheet: ruleset*

ruleset: selector[,selector]* '{' [declaration ';']* '}'

declaration: property ':' expr ['! important']?

```
h1 {color: blue;}
```

8

Selectors

- Type E
- Universal *
- Grouping E,G,F
- Attribute E[foo="hi"]
- ID #myID or E#myID
- Class .myClass
- Pseudo-element E:pseudo-element
- Contextual
 - Descendent E F
 - Child E > F
 - Adjacent E + F

9

Element Selector

```
h1 {  
  border: solid red 1;  
  margin: 1cm;  
  padding: 1em;  
  width: 40%;  
  text-align: left;  
}
```

10

Grouping

- One can group multiple headings, and apply a set of styles to all.

```
h1, h3 {  
  color: blue;  
  font-family: helvetica  
}
```

11

Attribute as Selector

- Use attribute values to conditionally apply style

– XHTML

```
<li type="grad">M.Sc.</li>
```

```
<li type="undergrad">B.Sc.</li>
```

– CSS

```
li[type="undergrad"] { color: black;}
```

```
li[type="grad"] { color: blue;}
```

12

ID as Selector

- XHTML allows any element to have a special "id" attribute that is unique in the document.
 - Can be used as the target for a hyperlink
 - Can be used to associate style properties with a particular element
- XHTML

```
<li id="i1">Programming the WWW</li>
```
- CSS

```
li#i1      { color: blue; }
li[id="i1"]
```

13

Class as Selector

- Use class to conditionally apply style
 - XHTML

```
<tr class="header"> ...
```
 - CSS

```
tr.header      { color: blue; }
```

14

Typographical Pseudo-Elements

- Hack for typographically important regions of text that are not delimited.
 - First-line formatting

```
P:first-line { font-weight:bold }
```
 - First-letter (drop-caps) formatting

```
P:first-letter { font-size: 200%; float: left }
```

This", quote I, is a paragraph that could be split anywhere in this sentence.

15

Anchor Pseudo-Classes

- A hack to account for elements whose behaviour changes through time.

```
a:link { color: green } /* unvisited link */
a:visited { color: purple } /* visited link */
a:active { color: red } /* active link */
```

 - No effect on elements other than <a>, so 'a' may be omitted.

16

Contextual Selectors

- CSS can match a search pattern on a stack of open elements.
- Siblings.

```
p+p      { font-size : 24pt; }
```
- Parents, ancestors, etc.

17

Inheritance

- Most styles are inherited into nested elements.
- Can set a "default" style by setting property values for the <body> element.

18

The Cascade

- >1 style sheet can influence the presentation simultaneously.
 - modularity
 - Web site, specific Web page, user, browser
- For different properties, all matching selectors are applied
- In the case of a property conflict
 - each rule is assigned a weight
 - heaviest wins

19

Weight Algorithm

1. Find all declarations that apply to the element/property
 - if none, then inherit
 - if no inheritance, use initial value
2. Sort by presence of 'important'
3. Sort by origin (author,user,browser)
 - author wins over user, user wins over browser default
4. Sort by specificity of selector
 - more specific wins ('h1 p' wins over 'p')
5. Sort by order specified
 - latter specified wins

20

Available Formatting

- Display
- Box
- Font
- Text
- Floating elements

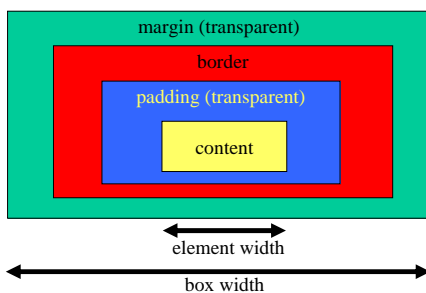
21

Display

- none | block | inline | list-item | table
- none: turns off display
- block: opens a new box (new-line before and after)
- inline: continues within same line
- list-item: same as block with a list-item marker
- table: block table

22

Box Formatting Model



23

Box Properties

- margin-top, margin-bottom, margin-left, margin-right, margin
- padding-top, padding-bottom, padding-left, padding-right, padding
- border-top-width, border-bottom-width, border-right-width, border-left-width

24

Font Families

- font-family: [family-name | generic-family] [, [family-name | generic-family]]*
 - e.g.,
 - body { font-family: gill, helvetica, sans-serif }
 - generic-families:
 - serif .This is a serif font
 - sans-serif .This is a sans-serif font
 - cursive .This is a cursive font
 - fantasy .This is a fantasy font
 - Monospace .This is a monospace font
 - .This is a haettenschweiler font
- Example: [fontcss.html](#)

25

Font Properties

- font-style: normal | italic | oblique
 - italic matches if keyword italic or oblique found
 - else must match exactly
- font-variant: normal | small-caps
 - small-caps satisfied if keyword present or can be synthesized
- font-weight: normal | bold | bolder | lighter | 100-900
 - always matches
- font-size: absolute | relative | length | percentage
 - matches within UA-defined tolerance
 - scalable fonts matched to within pixel
 - bitmapped fonts matched within as much as 20%

26

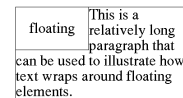
Text Properties

- word-spacing: normal | length
- letter-spacing: normal | length
- text-decoration: underline | overline | line-through | blink
- text-transform: capitalize | uppercase | lowercase | none
- text-align: left | right | center | justify
- text-indent: length | percentage
- line-height: number | length | percentage
- Example: [textcss.html](#)

27

Floating Elements

- float: left | right | none
 - left, right: formatted as block
 - left: moved to the left, text wraps on the right
 - right: moved to the right, text wraps on the left
- Example: [floatcss.html](#)



28

Vertical Alignment

- vertical-align: baseline | sub | super | top | text-top | middle | bottom | text-bottom | percentage
- Example: [textvaligncss.html](#)

$E = mc^2$ This famous equation was first discovered by Einstein. It relates the mass of matter to its energy content.

29

Generated Text

- Add text content before or after XML data
- Pseudo-elements
 - :before
 - :after
- XHTML
 - Sebesta
- CSS
 - li:before {content: "Author's Name -";}
- Browser
 - Author's Name - Sebesta

30