CSS, XSL & XSLT

ESA 2016/2017

Adam Belloum

a.s.z.belloum@uva.nl

Material Prepared by Eelco Schatborn

The birth of CSS

- **HTML grew**, more and more stylistic capabilities, became **more complex** to **write** and **maintain**
- consistent site **appearance** difficult because of different browser implementations, also lacked user control

- introducing CSS
  - 9 different style sheet languages were proposed to the W3C
  - 2 were chosen as the basis for CSS, CHSS and SSP
  - CSS version 1 become an **official W3C Recommendation in December 1996**

Definition

- Cascading Style Sheets (CSS) **form** the **presentation layer** of the user interface.
- Tells the browser agent ***how*** **the element** is to be **presented** to the user.

Why CSS?

- CSS **removes** the **presentation attributes** from the structure allowing reusability, ease of maintainability, and an **interchangeable** presentation layer.

- HTML was never meant to be a presentation language. **Proprietary vendors** have created tags to add presentation to structure.
  - <font>
  - <b>
  - <i>

- CSS allows us to make **global** and **instantaneous** changes easily.

Cascading styles

- The site designer has more
  - control on the style
  - easily maintained
- The document style can be influenced by multiple style sheets
- sheet can **inherit** or **“cascade"** from another

Benefit of CSS

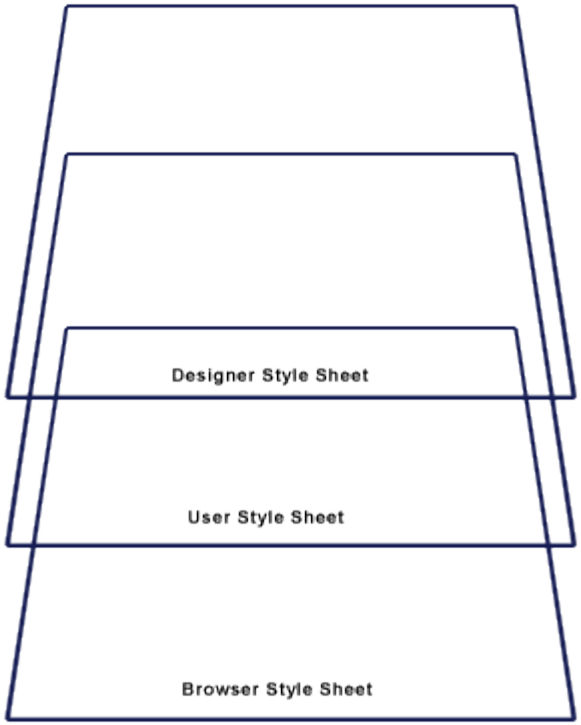
- Powerful and **flexible** way to specify the formatting of HTML elements
- Share Style Sheets **across multiple documents** or entire Web Site
- Rules are applied in a **hierarchical** manner (precedence rules)

http://www.techrepublic.com/article/css-101-handling-multiple-rules-for-the-same-element/6164216

The Cascade

- The power of CSS is found in the "cascade" which is the **combination** of the
  - **browser's default styles**,
  - **external style sheets**,
  - **embedded**,
  - **inline**,
  - and even **user-defined styles**.

- The cascade sets priorities on the individual styles which effects **inheritance**.



CSS Inheritance

- Allows elements to "**inherit**" styles from parent elements.
- **Reduces** the amount of CSS to set styles for child elements.

- Unless a more specific style is set on a **child** element, the element looks to the **parent** element for its styles.

Using Style Sheets

- External Style Sheet

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

  - Also a "media" descriptor (screen, print, etc)
  - Preferred method.

- Embedded Styles

```
<style type="text/css">
body {}
</style>
```

- Inline Styles

```
<p style="font-size: 12px">Lorem ipsum</p>
```

CSS version History

- CSS 1

  become an official **W3C Recommendation** in December 1996, in **most** browsers from 2000 onwards

- CSS 2

  published as a W3C Recommendation in May 1998, **still not fully** supported in browsers

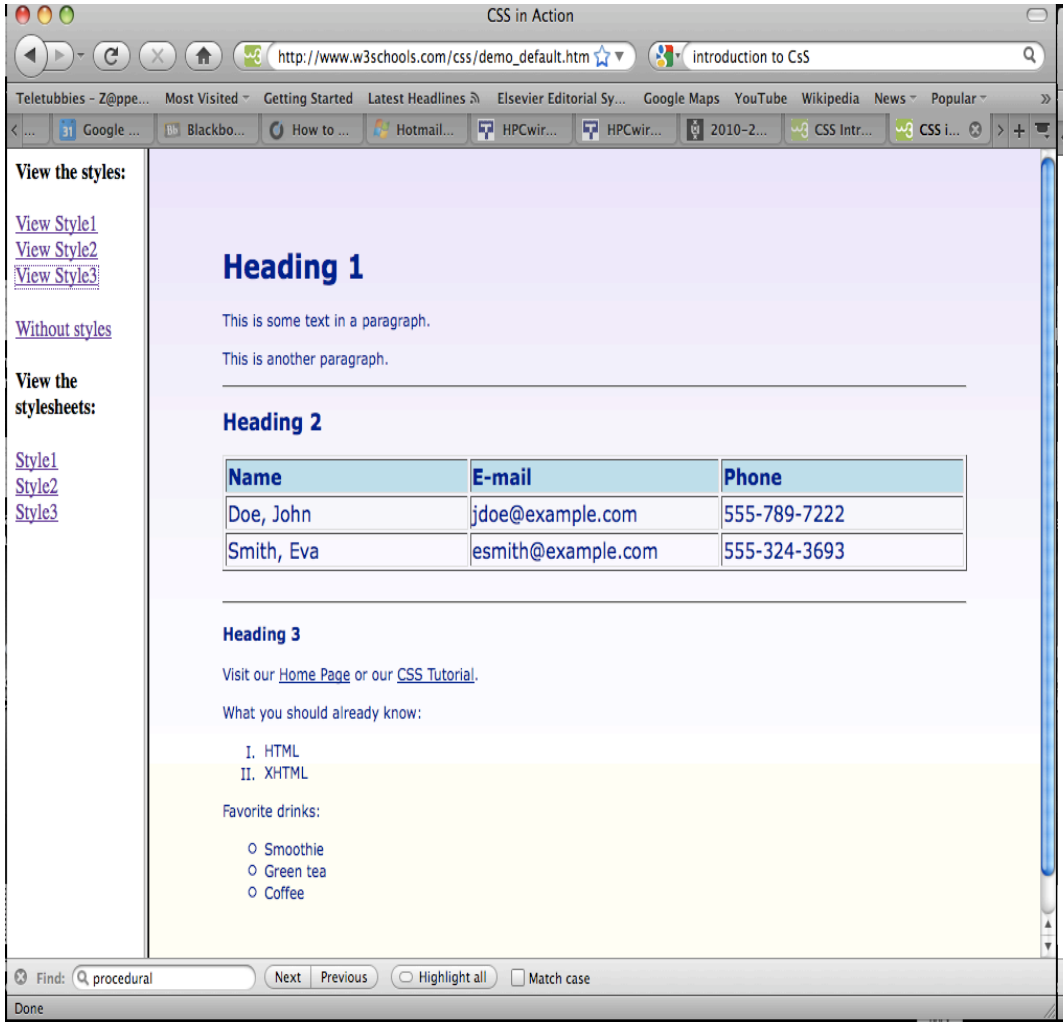
- CSS 2 revision 1,

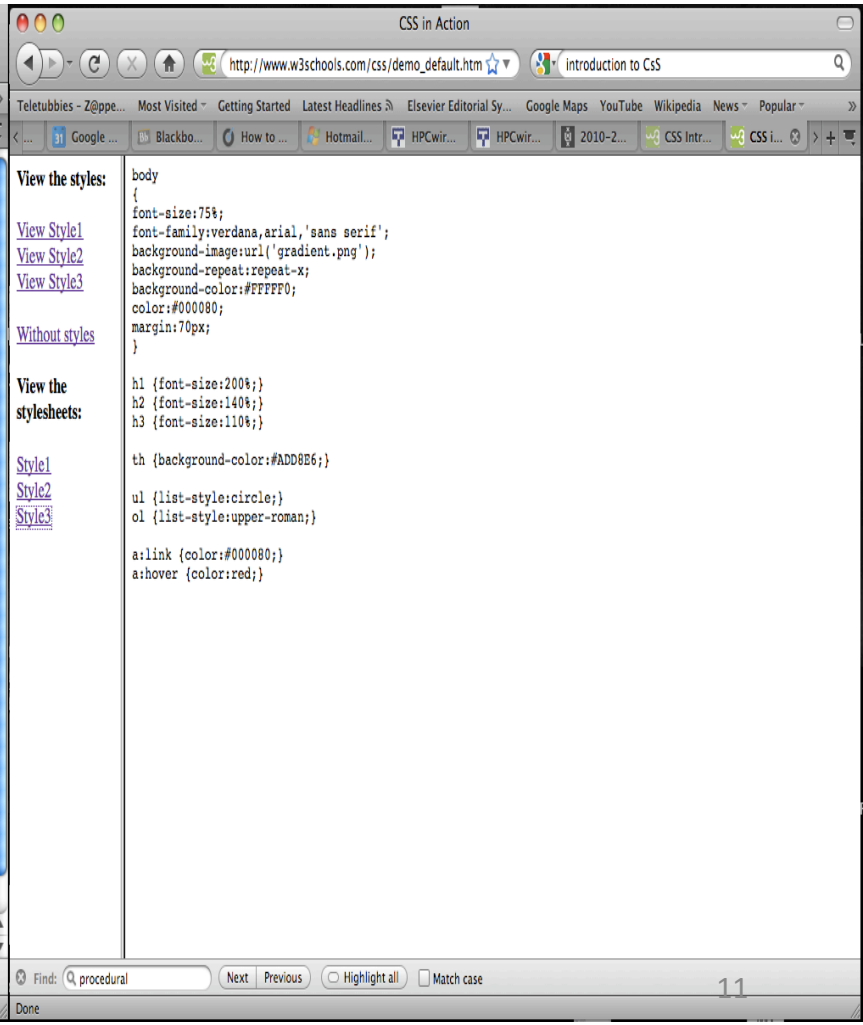
  June 2005, fixes errors in CSS 2, **removes poorly-supported features**, adds features already supported in browsers

- CSS 3.0

  currently under development, CSS 3 is modularized and will consist of several separate Recommendations

CSS in action





CSS adapt the layout to the output Media

Cascading Style Sheets (CSS)

- a stylesheet language used to describe the **presentation** **of a document** written in a **markup language** to style web pages written in HTML and XHTML
- Can be applied to any **XML document** (SVG, XUL,...)
- **separation** **of** **content** **and** **presentation**

```
Selector {property: value; property: value;
property: value;}
```

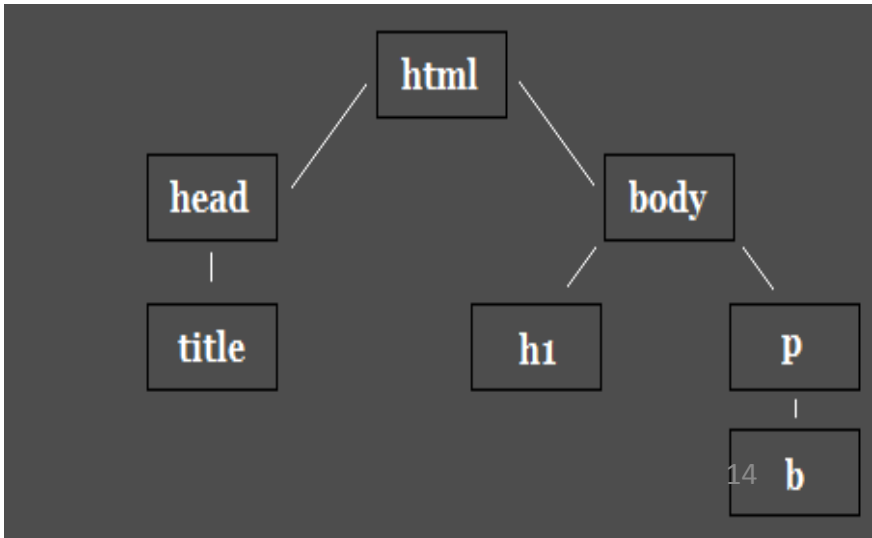
Example

Document Tree

- Every HTML, XHTML document is a document tree.

Below is an example of how a document tree looks.

```
<html>
    <head>
        <title>Document Tree</title>
    </head>
    <body>
        <h1>Header</h1>
        <p>Content inside a paragraph</p>
    </body>
</html>
```



http://www.wctutorials.com/reference/css/tree.html

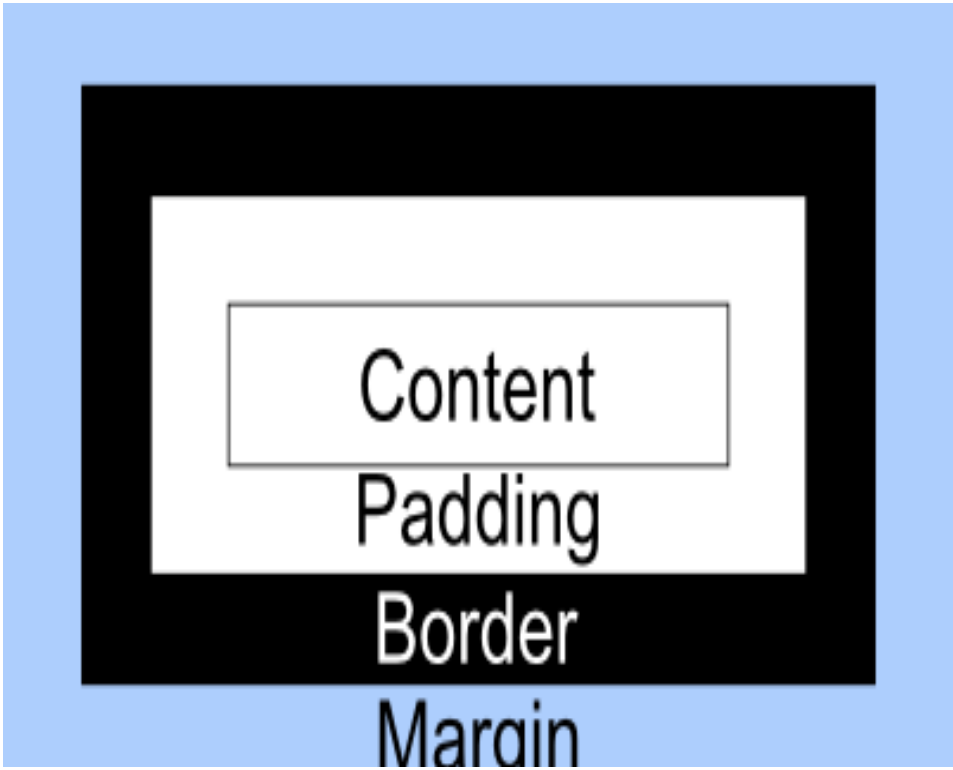
CSS Box Model Diagram

In CSS, the term "box model" is used when talking about **design and layout**.

CSS box model is a box that **wraps around HTML elements,** and it consists of: margins, borders, padding, and the actual content.

For example box model allows to add:

- a border around elements,
- and to define space between elements.



Multiple attributes

You can provide multiple **properties** to a **selector**:

```
p {
  font-weight: bold;
  color: yellow;
  background-color: black;
}
```

Grouping

You can specify arguments for multiple **selectors**

Example

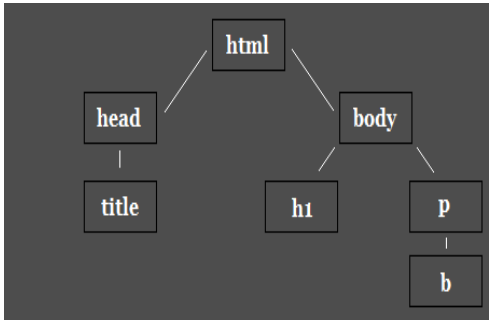
```
h1 { color: yellow; }
h2 { color: yellow; }
h3 { color: yellow; }
```

is equivalent to:

```
h1,h2,h3 { color: yellow; }
```

Selectors

Selectors are **patterns** used to **match elements** in the **document tree**.



- **Type** selectors `'E'`
  → matches any E element
- **Descendant** selectors `'E F'`
  → matches any F element that is a **descendant** of an E element
- **Child** selectors `'E > F'`
  → matches any F element that is a **child** of an element E
- **Adjacent** selectors `'E + F'`
  → matches any F element **immediately preceded** by an element E
- **Universal** selector `'*'`
  → matches any element

Type selectors

- matches the **name** of a document language **element type**
- matches **every instance** of the element type in the document tree
- element names are **case-sensitive** if the document language is

Example

```
h1 { color: red; }
```

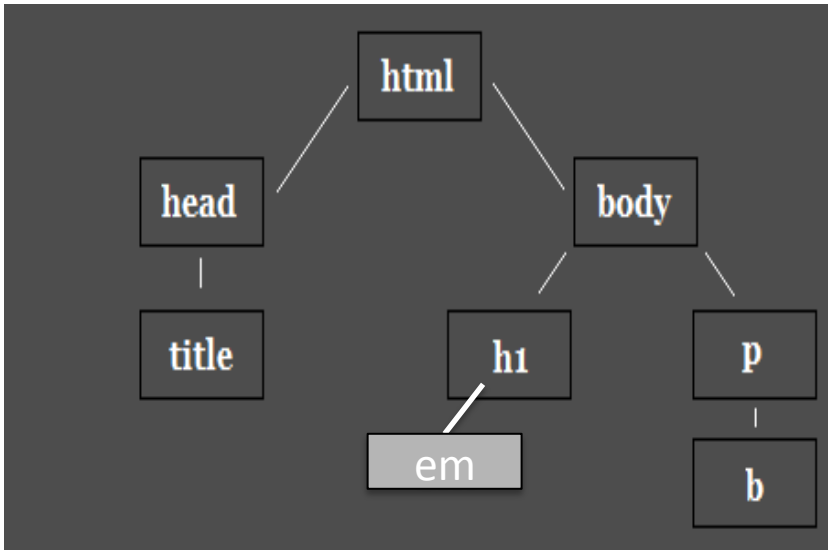
Matches **all h1** elements in the document tree

Descendant selectors

- match an element that is **the descendant of another element** in the document tree

Example

```
h1 { color: red }
em { color: red }
h1 em{ color: blue }
```



The third rule will match the EM in the following fragment:

```
<h1>This <span class="myclass">headline
is <em>    </em> important</span></h1>
```

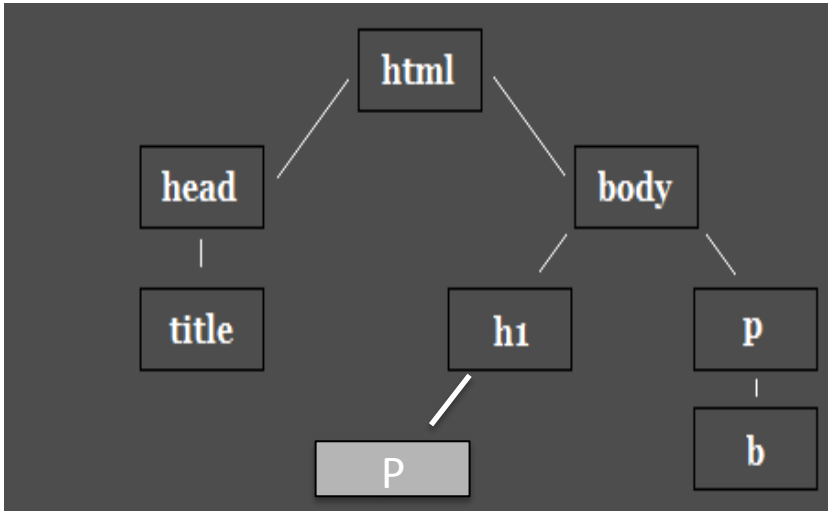
Child selectors

- match when an element is the child of some element
- descendant and child selectors can be mixed

Example

```
body > p {line-height:1.3}
```

matches all p elements that are children of **body**

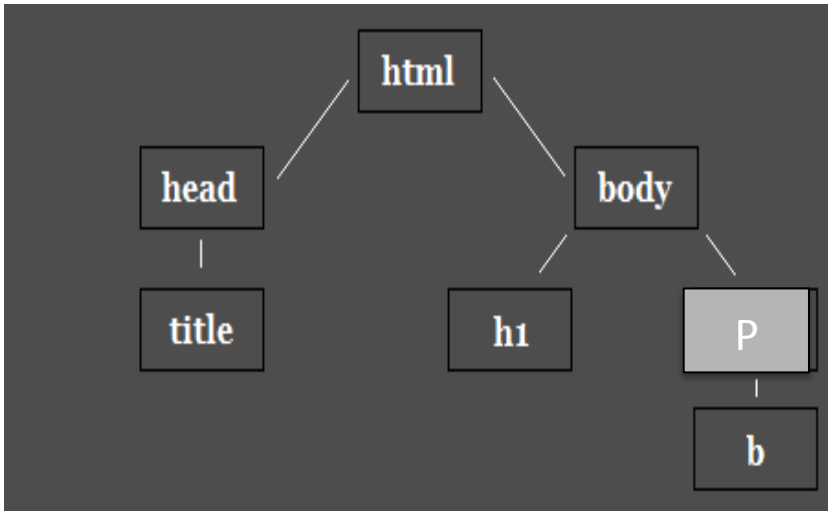


Example

```
div ol > li p {line-height:1.3}
```

Adjacent selectors

- match when an element is immediately preceded by some element



Example

reduce the vertical space separating an h1 and an <p> immediately Following h1

```
h1 + p { margin-top: -5mm }
```

```
<h1>Title</h1>
<p>Paragraph example.</p>
<p>Paragraph example.</p>
<p>Paragraph example.</p>

<div class="box">
  <p>Paragraph example.</p>
  <p>Paragraph example.</p>
</div>
```

Universal selector (`*')

- matches the name of any element type

Example

```
h1 * { color : red }
```

changes the color of all descendants of h1

CSS Comments

You can add comments to CSS source files by using the familiar `/*' and `*/' tokens:

```
/* This is some comment */
p
{
text-align: right;
/* This is another comment */
color: black;
font-family: arial
}
```

CSS Colors

- W3C Standard Color Names

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

- Hexadecimal RGB Values

  `#FF0000`

- RGB values

  `rgb(255,0,0)`

Examples

```
EM { color: red }
EM { color: rgb(255,0,0) }
```

More selectors

- **Attribute** selectors
  match **elements** by **attributes** defined in the source document
- **class** selectors
  an alternative notation when matching on the `class' attribute
- **ID** selectors
  match **elements** by **ID**
- **Pseudo classes**
  classify elements on characteristics other than their name, attributes or content
- **Pseudo elements**
  match abstractions in the document tree beyond those specified by the document language

Attribute selectors

- `E[foo]`

  Matches any E element with the "**foo" attribute** set (whatever the value)

- `E[foo="warning"]`

  Matches any E element whose "**foo" attribute** value is exactly equal to "warning"

- `E[foo~="warning"]`

  Matches any E element whose "**foo" attribute** value is a list of space-separated values, one of which is exactly equal to "warning"

- `E[lang|="en"]`

  Matches any E element whose "**lang" attribute** has a hyphen-separated list of values beginning (from the left) with "en"

http://www.w3.org/TR/CSS2/selector.html

Defining Style Classes

- To define an **element Style class** proceed the html element by a period and class name

  //define and abstract paragraph type

```
p.abstract { margin-left: 0.5in;
             margin-right: 0.5in;
             font-style:italic }
```

- To use supply the name of the style class in the **CLASS attribute** in the HTML element

```
<h1> New Advances in Physics </h1>
<p class="abstract">
Text </p>
```

Class selectors

- used with HTML
- Authors use the dot (.) notation when matching on the "class" attribute
- the attribute value must immediately follow the .

Example

The following assigns style only to H1 elements with class "sne":

```
h1.sne { color: red } /* h1[class~=sne] */
```

results in:

```
<h1>Not red</h1>
<h1 class="sne">Very red</h1>
```

ID selectors

- Document may contain **attributes** that are of **type ID**.
- ID attributes are know to be unique
- **ID attribute** can be used to uniquely identify its element in HTML all ID attributes are named `id'

Example

```
h1#chapter1 { text-align: center }
```

```
<h1 id="chapter1"> …
```

ID vs. Class

- Identifier or class? What's the difference?
  - An **identifier** is specified only **once on a page** and has a **higher** inheritance specificity than a **class**.
  - A class is **reusable** as many times as needed in a page.

Pseudo classes (1)

- to permit formatting based on **information** that **lies outside** the document tree
  - do **not appear** in the document source or document tree.
  - **may only appear** after the subject of the selector.
- are allowed anywhere in selectors
- names are **case-insensitive**.

Pseudo classes (2)

- `E:link`
  matches element (hyperlink) E if it is not yet visited
- `E:visited`
  matches element (hyperlink) E if it is already visited
- `E:active`
  matches E during certain user actions
- `E:hover`
  matches E during certain user actions
- `E:focus`
  matches E during certain user actions
- `E:lang(|="en")`
  matches any E element whose "`lang`" attribute has a hyphen-separated list of values beginning with "`en`".

Pseudo elements (3)

- `E:first-line`

  the first formatted line of a paragraph

- `E:first-letter`

  may be used for "initial caps" and "drop caps"

- `E:before`

  can be used to insert generated content before an element's content

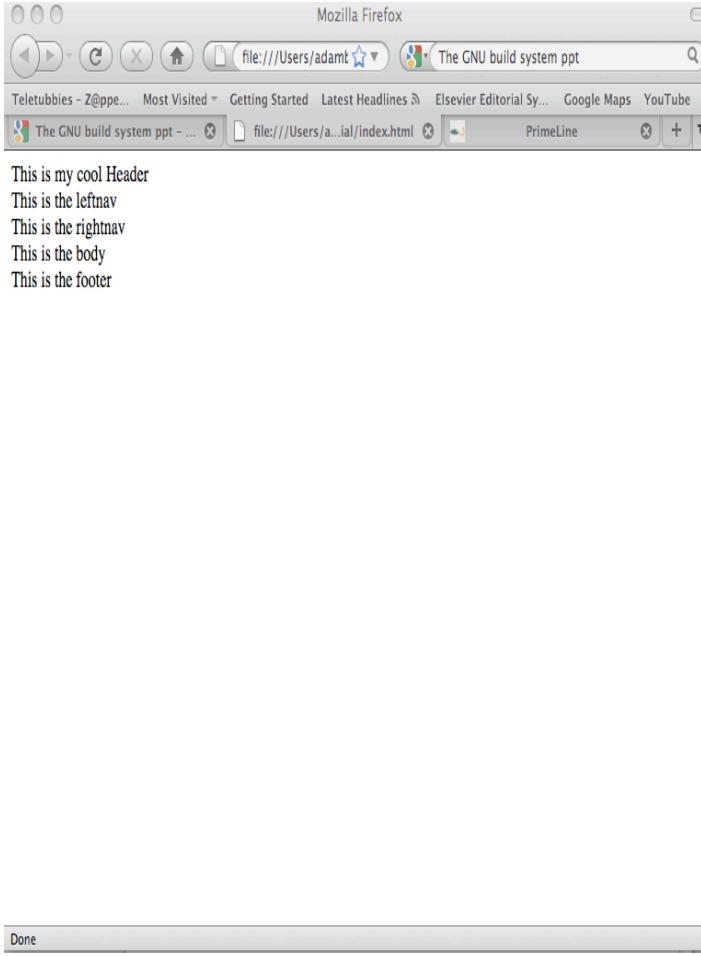
- `E:after`

  can be used to insert generated content after an element's content

CSS validators

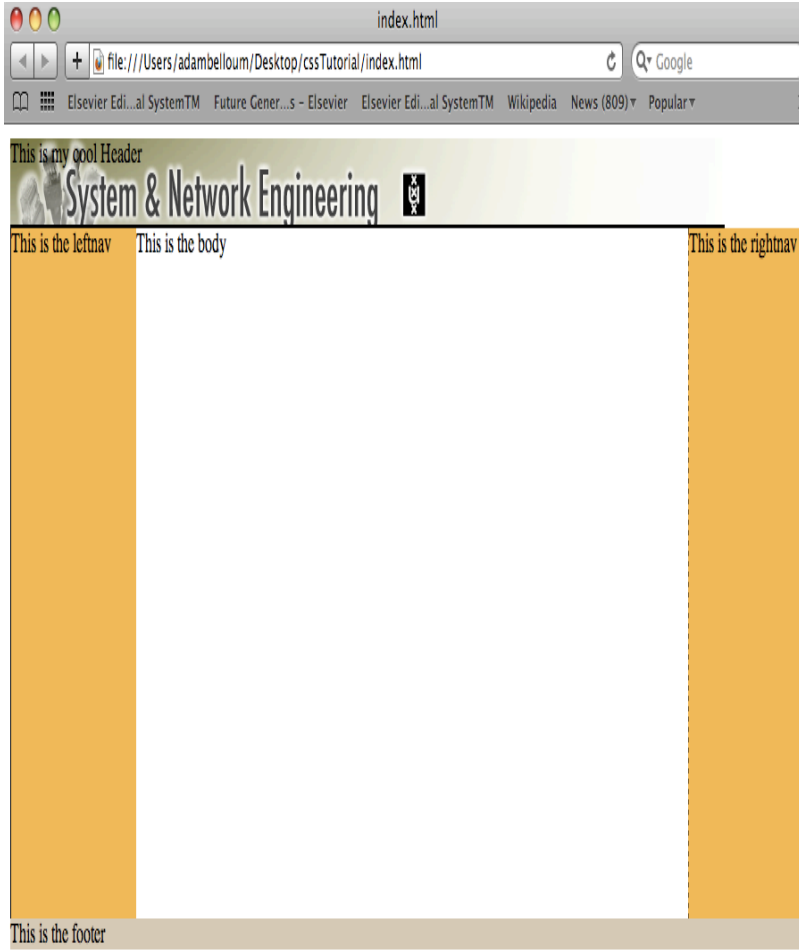
- CSSTidy
- W3C validators
- W3schools
- Acid3

```
<html>
<head>
</head>
<body>

<div id="header">This is my cool
   Header</div>
<div id="leftnav">This is the
   leftnav</div>
<div id="rightnav">This is the
   rightnav</div>
<div id="body">This is the body</div>
<div id="footer">This is the footer</
   div>

</body>
</html>
```

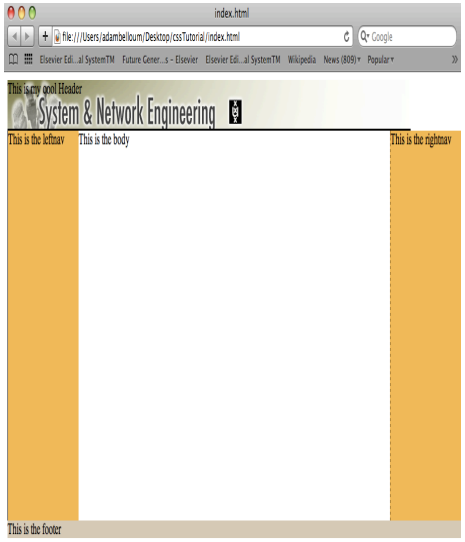


Try to create the following page

```
<html>
<head>
<link href="style.css" rel="stylesheet"
   type="css/text">
</head>
<body>
<div id="container">
<div id="header">This is my cool
   Header</div>
<div id="leftnav">This is the leftnav</
   div>
<div id="rightnav">This is the
   rightnav</div>
<div id="body">This is the body</div>
<div id="footer">This is the footer</
   div>
</div>
</body>
</html>
```

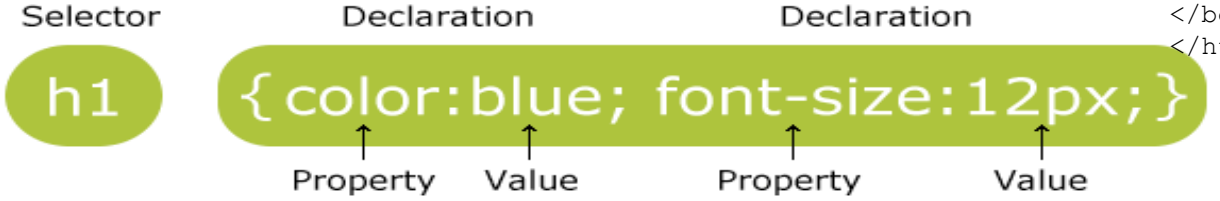


Some elements used to generate the previous layout

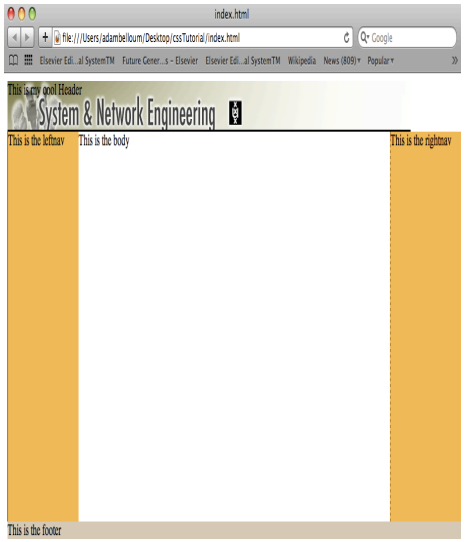


- `width: 900px;`
- `height: 50px;`
- `background-image: url(images/SNELogo.png);`
- `border-bottom: 2px solid #000000;`
- `border-left: 1px dashed #694717;`
- `float: left;`
- `float: right;`
- `background-color: #f8AA3c;`
- `border-left:1px solid #000000;`
- `clear: both;`

```
<html>
<head>
</head>
<body>

<div id="header">This is my cool Header</
    div>
<div id="leftnav">This is the leftnav</
    div>
<div id="rightnav">This is the rightnav</
    div>
<div id="body">This is the body</div>
<div id="footer">This is the footer</div>

</body>
</html>
```



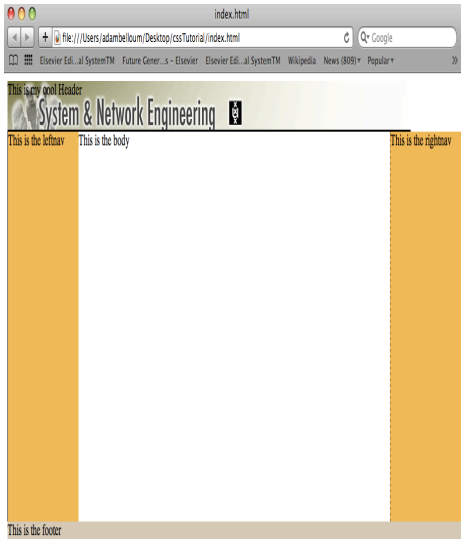
The HTML code




```
<html>
<head>
<link href="style.css" rel="stylesheet" type="css/text">
</head>
<body>
<div id="container">
<div id="header">This is my cool Header</div>
<div id="leftnav">This is the leftnav</div>
<div id="rightnav">This is the rightnav</div>
<div id="body">This is the body</div>
<div id="footer">This is the footer</div>
</div>
</body>
</html>
```

http://www.youtube.com/watch?v=H3KESBQTD8k&feature=channel

The CSS code




```
#header {

width: 800px;
height: 50px;
background-image: url(images/
   SNELogo.png);
border-bottom: 2px  solid #000000;

}

#leftnav  {

float:  left;
width: 140px;
height: 400px;
background-color: #f8AA3c;
border-left:1px solid #000000;
}
```

```
#rightnav {

float: right;
width: 140px;
height: 400px;
background-color: #F8AA3c;
border-left: 1px dashed #694717;

}

#body {

width: 620px;

}

#footer  {

clear: both;
background-color:  #D1C0A7;

}
```

http://www.youtube.com/watch?v=H3KESBQTD8k&feature=channel

Content

- Last lecture
  - XML & XHTML
- Today
  - CSS
  - XSL & XSLT

Extensible Stylesheet Language (XSL)

- family of transformation languages (`XSLT, XSL-FO, XPath`)
- Used to describe how to **format** or **transform** files encoded in **XML**
- use **valid XML** syntax
- can produce **HTML**, **plain-text**, or **PDF** among others …

Example

```
<xsl:if test="@author='Jones'">
  Hello Mrs. Jones!
</xsl:if>
```

XSL Transformations (XSLT)

- XML-based language
- **transformation** of XML documents into other XML or "human-readable" documents
- uses **template** rule processing, based on matching
- many implementations available
- browsers supporting transformation of XML to HTML through XSLT
  - Internet Explorer (MSXML engine)
  - Firefox, Mozilla, and Netscape (TransforMiiX engine)
  - Opera (native engine)
  - . . .

History

- XSLT 1.0
  - developed by the World Wide Web Consortium (W3C)
  - Originally part of the XSL development effort (1998-1999)
  - XSLT 1.0 was published as a Recommendation by the W3C on 16 November 1999
- XSLT 2.0
  - built in 2002-2006
  - based on richer data model and type system based on XMLSchema
  - reached W3C recommendation status on 23 January 2007

Templates

if a template is matched using a **pattern**, the rule applies

Example

XML source

```
<person>
  <name>Eelco</name>
</person>
<person>
  <name>Jaap</name>
</person>
```

Output

```
<li> Eelco </li>
<li> Jaap </li>
```

Example XSLT template

```
<xsl:template match="person">
<li> <xsl:value-of select="name"/> </li>
</xsl:template>
```

XSLT elements

Various elements usage:

```
<xsl:element attributes/options />
```

- conditional

  `if, choose, otherwise, when,...`
- iteration

  `for-each,...`
- template

  `apply-template, call-template, template,...`
- node

  `value-of, copy, copy-of, sort,...`
- attribute

  `attribute, attribute-set,...`

http://www.w3schools.com/xsl/xsl_w3celementref.asp XSLT version 1.0

```
<?xml version="1.0" encoding="UTF-8"?>
<catalog>
    <cd>
        <title>Empire Burlesque</title>
        <artist>Bob Dylan</artist>
        <country>USA</country>
        <company>Columbia</company>
        <price>10.90</price>
        <year>1985</year>
    </cd>
    <cd>
        <title>Hide your heart</title>
        <artist>Bonnie Tyler</artist>
        <country>UK</country>
        <company>CBS Records</company>
        <price>9.90</price>
        <year>1988</year>
    </cd>

...
</catalog>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
xmlns:xsl=
http://www.w3.org/1999/XSL/Transform>
<xsl:template match="/">
 <html>
 <body>
 <h2>My CD Collection</h2>
 <table border="1">
  <tr bgcolor="#9acd32">
   <th>Title</th>
   <th>Artist</th>
  </tr>
  <xsl:for-each select="catalog/cd">
  <tr>
   <td><xsl:value-of select="title"/></td>
   <td><xsl:value-of select="artist"/></td>
  </tr>
  </xsl:for-each>
 </table>
 </body>
 </html>
</xsl:template>
</xsl:stylesheet>
```

http://www.w3schools.com/xsl/xsl_for_each.asp

XSLT stylesheet (XML to XHTML) (1)

Initial XML

```
<?xml version="1.0" ?>
<list>
    <person username="JS1”>
      <name>John</name>
      <family_name>Smith</
      family_name>
    </person>
    <person username="MI1">
      <name>Morka</name>

      <family_name>Ismincius
      </family_name>
    </person>
</list>
```

XHTML output

```
<?xml version="1.0"
   encoding="UTF-8"?>
   <html xmlns="http://
   www.w3.org/1999/xhtml">
<head> <title>Testing XML
   Example</title> </head>

<body>
<h1>list</h1>
    <ul>
      <li>Ismincius, Morka</li>
      <li>Smith, John</li>
    </ul>
</body>
</html>
```

XSLT stylesheet (XML to XHTML) (2)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                                                    version="1.0”>

<xsl:template match="list">
 <html>
 <head> <title>Testing XML Example</title> </head>
 <body>
 <h1>list</h1>
 <ul>
 <xsl:apply-templates select="person">
 <xsl:sort select="family_name" />
 </xsl:apply-templates>
 </ul>
 </body>
 </html>
 </xsl:template>

<xsl:template match="person">
   <li>
    <xsl:value-of select="family_name"/>,
    <xsl:value-of select="name"/>
   </li>
</xsl:template>
</xsl:stylesheet>
```

```
<?xml version="1.0" ?>
<list>
      <person username="JS1”>
          <name>John</name>
          <family_name>Smith</family_name>
      </person>

      <person username="MI1">
          <name>Morka</name>
          <family_name>Ismincius</family_name>
      </person>
</list>
```

```
<?xml version="1.0" encoding="UTF-8"?>
    <html xmlns="http://www.w3.org/1999/xhtml">
<head> <title>Testing XML Example</title> </head>

<body>
<h1>list</h1>
      <ul>
          <li>Ismincius, Morka</li>
          <li>Smith, John</li>
      </ul>
</body>
</html>
```

Assignment

- Use your XML data from the last assignment
- write an XSLT stylesheet that transforms your XML data to valid XHTML strict, which can be viewed from within a browser
- then write a CSS stylesheet for the XHTML output that at least defines colors for each characteristic in your data, and makes these change on mouse-over
- Note
  - all XHTML, XSLT and CSS must validate (show this in your log)
  - upload your files to the wiki, note that you can also use the <code> tag with coloring!

Suggested Reading

Zen CSS Garden

http://www.csszengarden.com/

http://www.brucelawson.co.uk/2004/zengarden/

- Others

http://www.cssplay.co.uk/menu/amazing

http://en.wikipedia.org/wiki/Comparison_of_layout_engines_(CSS)

http://www.w3.org/2002/03/csslayout-howto

Suggested Resources

- Examples
  - http://www-scf.usc.edu/~csci571/Special/xsl_examples.html

Others

- Wikipedia
- http://www.w3schools.com/xsl/
- http://www.w3.org/TR/xslt
- http://en.wikipedia.org/wiki/XSLT_elements