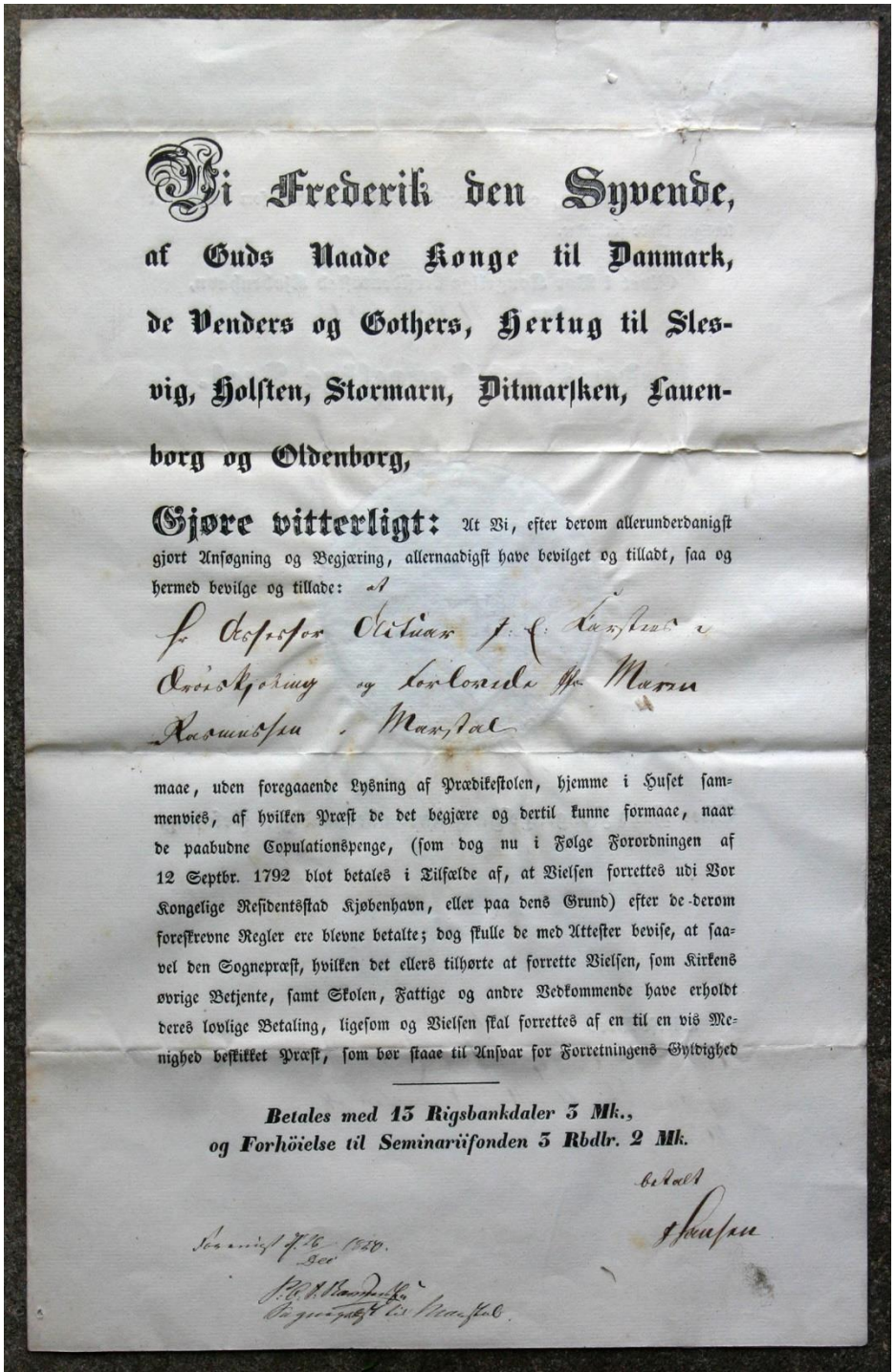
(Basic) Lecture 2

First of all

- <u>Content is the key word for html. It does not describe style!!</u>
- Scripts for behavior
- CSS for styling



Vi Frederik den Syvende,
af Guds Naade Konge til Danmark,
de Venders og Gothers, Hertug til Slesvig, Holsten, Stormarn, Ditmarsken, Lauenborg og Oldenborg,

Gjøre vitterligt: At Vi, efter derom allerunderdanigst gjort Ansøgning og Begjæring, allernaadigst have bevilget og tilladt, saa og hermed bevilge og tillade: at

maae, uden foregaaende Lysning af Prædikestolen, hjemme i Huset sammenvies, af hvilken Præst de det begjære og dertil kunne formaae, naar de paabudne Copulationspenge, (som dog nu i Følge Forordningen af 12 Septbr. 1792 blot betales i Tilfælde af, at Vielsen forrettes udi Vor Kongelige Residentsstad Kjøbenhavn, eller paa dens Grund) efter de derom foreskrevne Regler ere blevne betalte; dog skulle de med Attester bevise, at saavel den Sognepræst, hvilken det ellers tilhørte at forrette Vielsen, som Kirkens øvrige Betjente, samt Skolen, Fattige og andre Vedkommende have erholdt deres lovlige Betaling, ligesom og Vielsen skal forrettes af en til en vis Menighed beskikket Præst, som bør staae til Ansvar for Forretningens Gyldighed

Betales med 13 Rigsbankdaler 3 Mk., og Forhöielse til Seminariifonden 3 Rbdlr. 2 Mk.

betalt

HTML

- HTML is a markup language used for creating webpages. The contents of each webpage are defined by HTML tags. Basic page tags, such as <head>, <body>, and define sections of the page, while tags such as <table>, <form>, <image>, and <a> define elements within the page. Most elements require a beginning and end tag, with the content placed between the tags. For example, a link to the our favorite eclass home page may use the following HTML code:
- <a href="https://eclass.inha.ak.cr">Eclass</a>

Markup Language

- A markup language is a computer language that uses tags to define elements within a document. It is human-readable, meaning markup files contain standard words, rather than typical programming syntax.

Hypertext Markup Language (HTML)

- Describes the *content* and structure of information on a web page
- Not the same as the presentation (appearance on screen)
- Surrounds text content with opening and closing tags
- Each tag's name is called an element
  - syntax: <element> content </element>
  - example: This is a paragraph

XHTML

- Uses a markup format called XML
- XML + HTML = XHTML
- Standardized in 2000
- A strict XHTML page uses some different syntax and tags than HTML

Structure

To declare namespace and keep each name unique. For XHtml is mandatory

```
<!DOCTYPE html PUBLIC
"http://www.w3.org/TR/

<html xmlns="http://www.w3.org/1999/xhtml">
      <head>
            information about the page
      </head>

      <body>
            page contents
      </body>
</html>
```

HTML

- HTML is saved with extension .html
- Basic structure: tags that enclose content, i.e., elements
- **Header** describes the page
- **Body** contains the page's contents

Page Title <title>

```
…
        <head>
                <title> HARRY POTTER AND THE DEATHLY HALLOWS
- PART 2 </title>
        </head>
…                                                          HTML
```

- Placed within the head of the page
- Displayed in web browser's title mark and when bookmarking the page

Paragraph <p>

```
…
        <body>
                <p> Harry Potter and the Deathly Hallows,
the last book in the series, begins directly after the
events of the sixth book.
Voldemort        has completed his ascension to power and
gains        control of the Ministry of Magic</p>
        </body>
…
```

HTML

Harry Potter and the Deathly Hallows, the last book in the series, begins directly after the events of the sixth book. Voldemort has completed his ascension to power and gains control of the Ministry of Magic

output

- Placed within the body of the page

Headings <h1>, <h2>, … <h6>

```
<h1> Harry Potter </h1>
<h2> Books </h2>
<h3> Harry Potter and the Philosopher's Stone </h3>
```

HTML

Harry Potter

Books

Harry Potter and the Philosopher's Stone

output

Horizontal rule <hr />

```
<p> First paragraph </p>
<hr />
<p> Second Paragraph </p>
```

HTML

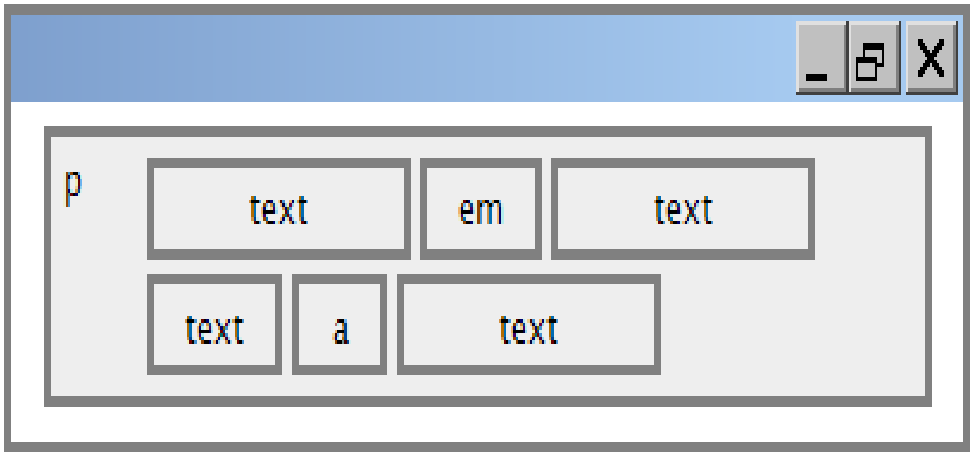
First Paragraph

Second Paragraph

output

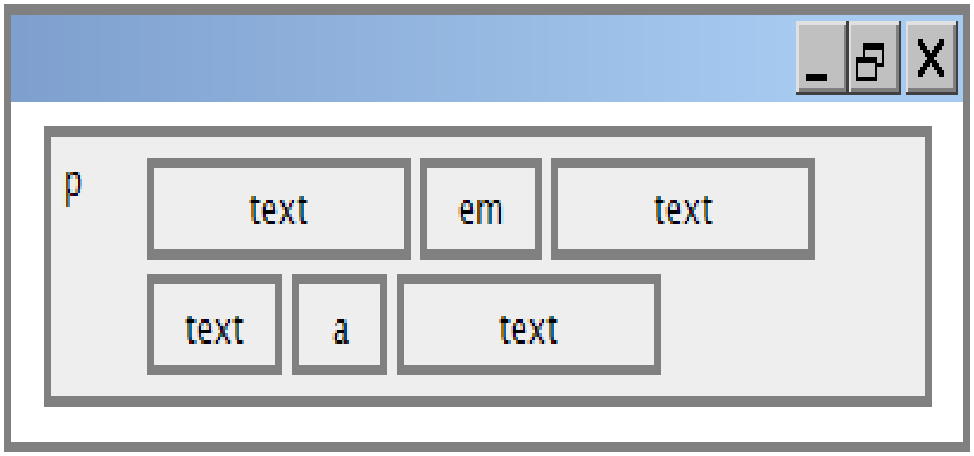
- Should be immediately closed with />

Block and Inline Statements



- Block elements contain an entire large region of content
  - examples: paragraphs, lists, table cells
  - the browser places a margin of whitespace between block elements for separation

Block and Inline Statements (cont.)



- Inline elements affect a small amount of content
  - examples: bold text, code fragments, images
  - the browser allows many inline elements to appear on the same line
  - must be nested inside a block element

More HTML tags

- Some tags can contain additional information called attributes
  - syntax: **<element attribute="value" attribute="value"> content </element>**
  - example: <a href="page2.html">Next page</a>

More HTML tags

- Some tags don't contain content; can be opened and closed in one tag
  - syntax:

<element attribute="value" attribute="value" />

  - example: **<hr />**
  - example:

<img src="Harry.jpg" alt="pic of Harry Potter" />

Links <a>

```
<p>
Search
<a href="http://www.google.com/">Google</a>
now!
</p>
```

HTML

Search Google now!

output

- The **href** attribute specifies the destination URL
- Links or *anchors* are inline elements, so they must be placed inside a block element such as a `p` or `h1`

More about anchors

```
<p><a href="deathlyHallows-book.html">Harry Potter and the
Deathly Hallows Book</a></p>

<p><a href="http://en.wikipedia.org"
title="Search">Wikipedia</a></p>
```

HTML

Harry Potter and the Deathly Hallows

Wikipedia

output

- Types of URLs that can appear in anchors:
  - Absolute: to another web site
  - Relative: to another page on this web site

Nesting tags

Bad

```
<p>
<a href=" deathlyHallows-book.html"> Harry Potter and the
Deathly Hallows Book </p>
<p>
This text also links to Harry Potter Book</a>
</p>
```

HTML

- Tags must be correctly nested: a closing tag must match the **most recently opened tag**
- The browser may render it correctly anyway, but it is invalid XHTML

Images <img>

```
<img src="images/tobby.jpg" alt="Tobby from Harry Potter" />
```
HTML

- The src attribute specifies source of the image URL
- XHTML also requires an alt attribute describing the image

More about images

```
<a href="http://harrypotter.net/">
<img src="images/dumbledore.jpg" alt="Dumbledore from Harry
Potter"
title="Alas! Ear wax!"/>
</a>
```
HTML

- If placed inside an a anchor, the image will become a link
- The title attribute specifies an optional tooltip

Line Break

```
<p>One Ring to rule them all, One Ring to find them, <br /> One Ring to bring them all and in the darkness bind them.</p>
<p> In the Land of Mordor where the Shadows lie. </p>
```

HTML

One Ring to rule them all, One Ring to find them,
One Ring to bring them all and in the darkness bind them

In the Land of Mordor where the Shadows lie.

output

- br should be immediately closed with />
- br should not be used to separate paragraphs or used multiple times in a row to create spacing

Comments <!-- ... -- >

```
<!-- My web page, by Bob Student
CSE 380, Fall 2048 -->
<p>CS courses are <!-- NOT --> a lot of fun!</p>
```

HTML

CS courses are a lot of fun!

output

- Comments are useful for disabling sections of a page
- Comments cannot be nested and cannot contain a --

Phrase elements ,

```
<p>
HTML is <em>really</em>,
<strong>REALLY</strong> fun!
</p>
```

HTML

HTML is *really* **REALLY** fun!

output

- **em**: emphasized text (usually in italic)
- **strong**: strongly emphasized text (usually in bold)
- The tags must be properly nested for a valid page

Unordered list: <ul>, <li>

```
<ul>
<li>No shoes</li>
<li>No shirt</li>
<li>No problem!</li>
</ul>
```

HTML

- No shoes
- No shirt
- No problem!

output

- **ul** represents a bulleted list of items (block)
- **li** represents a single item within the list (block)

More about unordered lists

```
<ul>
<li>Harry Potter characters:
<ul>
<li>Harry Potter</li>
<li>Hermione</li>
<li>Ron</li>
</ul>
</li>
<li>LOTR characters:
<ul>
<li>Frodo</li>
<li>Bilbo</li>
<li>Sam</li>
</ul>
</li>
</ul>
```

HTML

More about unordered lists (cont.)

- Harry Potter characters:
  - Harry Potter
  - Hermione
  - Ron
- LOTR characters:
  - Frodo
  - Bilbo
  - Sam

output

Ordered list <ol>

```
<p>Apple business model:</p>
<ol>
<li>Beat Microsoft</li>
<li>Beat Google</li>
<li>Conquer the world!</li>
</ol>
```

HTML

Apple business model:

1. Beat Microsoft
2. Beat Google
3. Conquer the world

output

- **ol** represents a numbered list of items
- we can make lists with letters or Roman numerals using CSS (later)

Common error: Not closing a list

```
<ul>
<li>No shoes</li>
<li>No shirt</li>
<li>No problem!</li>
<p>Paragraph after list...</p>
```
HTML

- No shoes
- No shirt
- No problem!

Paragraph after list...

output

- If you leave a list open, subsequent contents will be indented

Common Error: Improper nested list placement

```
<ul>
<li>Harry Potter characters:</li>
<ul>
<li>Harry Potter</li>
<li>Hermione</li>
<li>Ron</li>
</ul>
</li>
<li>LOTR characters:
<ul>
<li>Frodo</li>
<li>Bilbo</li>
<li>Sam</li>
</ul>
</ul>
```

HTML

- closing the outer li too early (or not at all) will render correctly in most browsers, but it is incorrect XHTML

Definition list <dl>, <dt>, <dd>

```
<dl>
<dt>newbie</dt> <dd>one who does not have mad skills</dd>
<dt>jaded</dt> <dd>tired, bored, or lacking enthusiasm
</dd>
<dt>frag</dt> <dd>a kill in a shooting game</dd>
</dl>
```

HTML

newbie
: one who does not have mad skills

jaded
: Tired, bored, or lacking enthusiasm

frag
: a kill in a shooting game

output

- **dl** represents a list of definitions of terms
- **dt** represents each term, and **dd** its definition

Tables <table>, <tr>, <td>

```
<table>
	<tr><td>1,1</td><td>1,2 okay</td></tr>
	<tr><td>2,1 real wide</td><td>2,2</td></tr>
</table>
```

HTML

1,1	1,2 okay
2,1 real wide	2,2

output

- `table` defines the overall table, `tr` each row, and `td` each cell's data
- Useful for displaying large row/column data sets
- NOTE: tables are sometimes used by novices for web page layout, but this is not proper semantic HTML and should be avoided

Table headers, captions: <th>, <caption>

```
<table>
        <caption>My important data</caption>
        <tr><th>Column 1</th><th>Column 2</th></tr>
        <tr><td>1,1</td><td>1,2 okay</td></tr>
        <tr><td>2,1 real wide</td><td>2,2</td></tr>
</table>
```

HTML

My important data

Column 1	Column 2
1,1	1,2 okay
2,1 real wide	2,2

output

- `th` cells in a row are considered headers
- a caption at the start of the table labels its meaning

Quotations <blockquote>

```
<p>As Lincoln said in his famous Gettysburg Address:</p>
      <blockquote>
      <p>Fourscore and seven years ago, our fathers
brought forth
      on this continent a new nation, conceived in
liberty, and
      dedicated to the proposition that all men are
created equal.</p>
</blockquote>
```

HTML

As Lincoln said in his famous Gettysburg Address:

Fourscore and seven years ago, our fathers brought forth on this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created equal.

output

- a lengthy quotation

Inline quotations <q>

```
<p>Quoth the Raven, <q>Nevermore.</q></p>
```
HTML

Quoth the Raven, “Nevermore.”

output

- a short quotation
- Why not just write the following?
- Quoth the Raven, "Nevermore."
- We don't use " marks for two reasons:
  - XHTML shouldn't contain literal quotation mark characters; they should be written as "
  - using <q> allows us to apply CSS styles to quotations

HTML Character Entities

character(s)	entity
< >	< >
é è ñ	é è ñ
™ ©	™ ©
π δ Δ	π δ Δ
И	И
" &	" &

Inline quotations <q>

```
&lt;p&gt;
&lt;a
href=&quot;http://google.com/search?q=xenia&amp;ie=utf-
8&amp;aq=t&quot;&gt;
Search Google for Xenia
&lt;/a&gt;
&lt;/p&gt;
```

HTML

 <a href="http://google.com/search?q=xenia&ie=utf-8&aq=t"> Search Google for Xenia </a>

output

Computer code <code>

```
<p>
The <code>ul</code> and <code>ol</code>
tags make lists.
</p>
```
HTML

The `ul` and `ol` tags make lists.

output

- code: a short section of computer code

Preformatted text <pre>

```
<pre>
Bill Gates speaks
    You will be assimilated
  Microsoft fans delirious
</pre>
```

HTML

```
Bill Gates speaks
    You will be assimilated
  Microsoft fans delirious
```

output

- Displayed with exactly the whitespace / line breaks given in the text
- Shown in a fixed-width font by default

Preformatted text <pre>

```
<pre><code>
        public static void main(String[] args) {
            System.out.println("Hello, world!");
        }
</code></pre>
```

HTML

```
public static void main(String[] args) {
            System.out.println("Hello, world!");
        }
```

output

- When showing a large section of computer code, enclose it in a `pre` to preserve whitespace and a `code` to describe the semantics of the content

Web Standards

- Why use XHTML and web standards?
  - more rigid and structured language
  - more interoperable across different web browsers
  - more likely that our pages will display correctly in the future
  - can be interchanged with other XML data: SVG (graphics), MathML, MusicML, etc.

W3C XHTML Validator

```
<p>
	<a href="http://validator.w3.org/check/referer">
	<img src="http://www.w3.org/Icons/valid-xhtml11"
alt="Validate" />
	</a>
</p>
```

HTML

- checks your HTML code to make sure it meets the official strict XHTML specifications

Web page metadata <meta>

```
<meta name="description"
content="Harry Potter Official Website." />
<meta name="keywords" content="harry potter, harry potter
and the deathly hallows, deathly hallows, ministry of
magic, resurrection stone, clock of invisibility" />
<meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1" />
```
HTML

- information about your page (for a browser, search engine, etc.)
- placed in the head of your XHTML page
- meta tags often have both the name and content attributes
  - some meta tags use the http-equiv attribute instead of name

meta element to aid browser / web server

```
<meta http-equiv="Content-Type"
content="type of document (character encoding)" />
<meta http-equiv="refresh"
content="how often to refresh the page (seconds)" />
</head>
```
HTML

- using the Content-Type gets rid of the W3C "tentatively valid" warning

<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />

- the meta refresh tag can also redirect from one page to another:

<meta http-equiv="refresh" content="5;url=http://www.bjp.com" />

  - why would we want to do this? (example)

meta element to describe the page

```
<head>
<meta name="author"
content="web page's author" />
<meta name="revised"
content="web page version and/or last modification date"
/>
<meta name="generator"
content="the software used to create the page" />
</head>
```
HTML

- many WYSIWYG HTML editors (FrontPage, PageMaker, etc.) place their names in the meta generator tag

meta element to aid search engines

```
<head>
<meta name="description"
content="how you want search engines to display your page"
/>
<meta name="keywords"
content="words to associate with your page (comma
separated)" />
</head>
```
HTML

- these are suggestions to search engines about how to index your page
- the search engine may choose to ignore them

The good, the bad and the… ugly!

```
<p>
<font face="Arial">Shashdot.</font>
News for <b>nerds!!</b> You will <i>never</i>, <u>EVER</u>
be
<font size="+4" color="red">BORED</font> here!
</p>
```

HTML

Slashdot. News for **nerds!!** You will never, EVER be BORED here!

output

- Tags such as b, i, u, and font are discouraged in strict XHTML
- Why is this bad?

Cascading Style Sheets (CSS)

- Describes the appearance, layout, and presentation of information on a web page
  - HTML describes **the content** of the page
- Describes *how* information is to be displayed, not *what* is being displayed
- Can be embedded in HTML document or placed into separate .css file

Basic CSS rule syntax

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

```
p, h1, h2 {
font-family: sans-serif;
color: red;
}
```
CSS

- A CSS file consists of one or more **rules**
- Each rule starts with a **selector**
- A selector specifies an HTML element(s) and then applies style **properties** to them
  - a selector of * selects all elements

Attaching a CSS file <link>

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

HTML

```
<link href="style.css" type="text/css" rel="stylesheet" />
<link href="http://www.google.com/uds/css/gsearch.css"
rel="stylesheet" type="text/css" />
```

HTML

- A page can link to multiple style sheet files
  - In case of a conflict (two sheets define a style for the same HTML element), the latter sheet's properties will be used

Embedding style sheets: <style>

```
<head>
<style type="text/css">
p { font-family: sans-serif; color: red; }
h2 { background-color: yellow; }
</style>
</head>
```

HTML

- CSS code can be embedded within the head of an HTML page
- B*ad style* and should be avoided when possible (why?)

Inline styles: the style attribute

```
<p style="font-family: sans-serif; color: red;">
This is a paragraph</p>
```

HTML

This is a paragraph

output

- Higher precedence than embedded or linked styles
- Used for one-time overrides and styling a particular element
- B*ad style* and should be avoided when possible (why?)

CSS properties for colors

```css
p {
color: red;
background-color: yellow;
}
```

CSS

This paragraph uses the style above *output*

property	description
color	color of the element's text
background-color	color that will appear behind the element

Specifying colors

```
p { color: red; }
h2 { color: rgb(128, 0, 196); }
h4 { color: #FF8800; }
```

CSS

This paragraph uses the first style above

This h2 uses the second style above.

This h4 uses the third style above.

output

- color names: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white (white), yellow
- RGB codes: red, green, and blue values from 0 (none) to 255 (full)
- hex codes: RGB values in base-16 from 00 (0, none) to FF (255, full)

Grouping styles

```css
p, h1, h2 {
color: green;
}
h2 {
background-color: yellow;
}
```

CSS

This paragraph uses the above style.

This h2 uses the above styles.

output

- A style can select multiple elements separated by commas
- The individual elements can also have their own styles

CSS comments /*…*/

```css
/* This is a comment.
It can span many lines in the CSS file. */
p {
color: red; background-color: aqua;
}
```
CSS

- CSS (like HTML) is usually not commented as rigorously as programming languages such as Java
- The // single-line comment style is NOT supported in CSS
- The <!-- ... --> HTML comment style is also NOT supported in CSS

CSS properties for fonts

property	description
font-family	which font will be used
font-size	how large the letters will be drawn
font-style	used to enable/disable italic style
font-weight	used to enable/disable bold style

Complete list of font properties (http://www.w3schools.com/css/css_reference.asp#font)

font-family

```css
p {
font-family: Georgia;
}
h2 {
font-family: "Courier New";
}
```

CSS

This paragraph uses the first style above.

This h2 uses the second style above.

output

- Enclose multi-word font names in quotes

More about font-family

```
p {
font-family: Garamond, "Times New Roman", serif;
}
```
CSS

This paragraph uses the above style.

output

- We can specify multiple fonts from highest to lowest priority
- Generic font names:
  - serif, sans-serif, cursive, FANTASY, monospace
- If the first font is not found on the user's computer, the next is tried
- Placing a generic font name at the end of your font-family value, ensures that every computer will use a valid font

font-size

```
p {
        font-size: 24pt;
}
```
CSS

This paragraph uses the style above.

output

- units: pixels (`px`) vs. point (`pt`) vs. m-size (`em`)

`16px`, `16pt`, `1.16em`

- vague font sizes: `xx-small`, `x-small`, `small`, `medium`, `large`, `x-large`, `xx-large`, `smaller`, `larger`
- percentage font sizes, e.g.: `90%`, `120%`

font-size

```
p {
        font-size: 24pt;
}
```
CSS

This paragraph uses the style above.

output

- `pt` specifies number of point, where a point is 1/72 of an inch onscreen
- `px` specifies a number of pixels on the screen
- `em` specifies number of m-widths, where 1 em is equal to the font's current size

font-weight, font-style

```
p {
font-weight: bold;
font-style: italic;
}
```
CSS

This paragraph uses the style above.

output

- Either of the above can be set to normal to turn them off (e.g. headings)

CSS properties for text

property	description
text-align	alignment of text within its element
text-decoration	decorations such as underlining
line-height, word-spacing, letter-spacing	gaps between the various portions of the text
text-indent	indents the first letter of each paragraph

Complete list of text properties (http://www.w3schools.com/css/css_reference.asp#text)

text-align

```
blockquote { text-align: justify; }
h2 { text-align: center; }
```

CSS

The Gollum's Quote

We wants it, we needs it. Must have the precious. They stole it from us. Sneaky little hobbitses. Wicked, tricksy, false!

output

- text-align can be `left`, `right`, `center`, or `justify`

text-decoration

```
p {
text-decoration: underline;
}
```
CSS

This paragraph uses the style above.

output

- can also be `overline`, ~~`line-through`~~, `blink`, or `none`
- effects can be combined:

```
text-decoration: overline underline;
```

The list-style-type property

```
ol { list-style-type: lower-roman; }
```
CSS

- **Possible values:**

  i. `none` : No marker

  ii. `disc` (default), `circle`, `square`

  iii. `Decimal`: 1, 2, 3, etc.

  iv. `decimal-leading-zero`: 01, 02, 03, etc.

  v. `lower-roman`: i, ii, iii, iv, v, etc.

  vi. `upper-roman`: I, II, III, IV, V, etc.

  vii. `lower-alpha`: a, b, c, d, e, etc.

  viii. `upper-alpha`: A, B, C, D, E, etc.

  x. `lower-greek`: alpha, beta, gamma, etc.

  others: hebrew, armenian, georgian, cjk-ideographic, hiragana…

Body styles

```
body {
font-size: 16px;
}
```

CSS

- Applies a style to the entire body of your page
- Saves you from manually applying a style to each element

Cascading Style Sheets

- Properties of an element *cascade* together in this order:
  - browser's default styles
  - external style sheet files (in a <link> tag)
  - internal style sheets (inside a <style> tag in the page's header)
  - inline style (the style attribute of the HTML element)

Inheriting styles

```
body { font-family: sans-serif; background-color: yellow;
}
p { color: red; background-color: aqua; }
a { text-decoration: underline; }
h2 { font-weight: bold; text-align: center; }
```

CSS

This is a heading

A styled paragraph. Previous slides are available on the website.

- A bulleted list

output

- when multiple styles apply to an element, they are inherited
- a more tightly matching rule can override a more general inherited rule

Styles that conflict

```
p, h1, h2 { color: blue; font-style: italic; }
h2 { color: red; background-color: yellow; }
```

CSS

This paragraph uses the first style above.

This heading uses both styles above.

output

- when two styles set conflicting values for the same property, the latter style takes precedence

W3C CSS Validator

```
<p>
<a href="http://jigsaw.w3.org/css-
validator/check/referer">
<img src="http://jigsaw.w3.org/css-validator/images/vcss"
alt="Valid CSS!" /></a>
</p>
```

CSS

output

- jigsaw.w3.org/css-validator/
- checks your CSS to make sure it meets the official CSS specifications

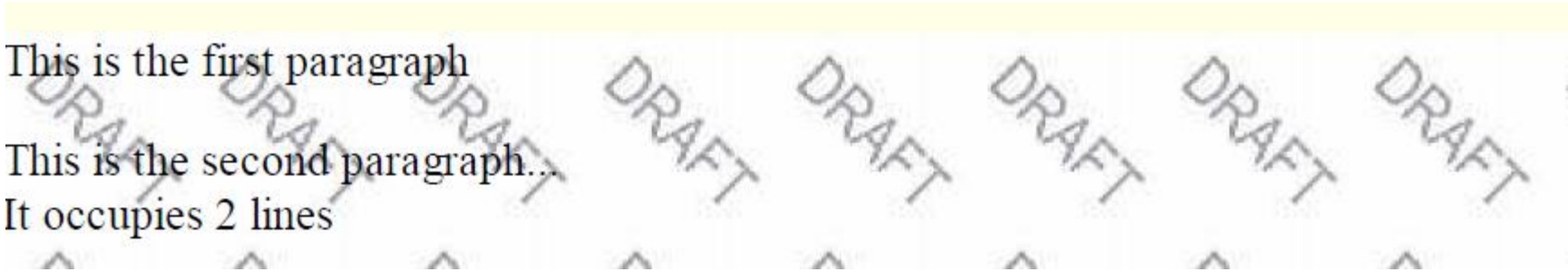
CSS properties for backgrounds

property	description
background-color	color to fill background
background-image	image to place in background
background-position	placement of bg image within element
background-repeat	whether/how bg image should be repeated
background-attachment	whether bg image scrolls with page
background	shorthand to set all background properties

background-image

```
body {
background-image: url("images/draft.jpg");
}
```

CSS



- background image/color fills the element's content area

background-repeat

```
body {
background-image: url("images/draft.jpg");
background-repeat: repeat-x;
}
```

CSS

- can be repeat (default), repeat-x, repeat-y, or no-repeat

background-position

```
body {
background-image: url("images/draft.jpg");
background-repeat: no-repeat;
background-position: 370px 20px;
}
```
CSS

This is the first paragraph

This is the second paragraph...
It occupies 2 lines

- value consists of two tokens, each of which can be top, left, right, bottom, center, a percentage, or a length value in px, pt, etc.
- value can be negative to shift left/up by a given amount

Aside: Favorites icon ("favicon")

```
<link href="filename" type="MIME type" rel="shortcut icon"
/>
```
HTML

```
<link href="yahoo.gif" type="image/gif" rel="shortcut
icon" />
```
HTML

- The link tag, placed in the HTML page's head section, can specify an icon
  - this icon will be placed in the browser title bar and bookmark/favorite

HTML id attribute

```
<p>Coding Horror! Coding Horror!</p>
<p id="mission">Our mission is to combine programming and
<q>human</q> factors with geekiness!</p>
```

HTML

Coding Horror! Coding Horror!

Our mission is to combine programming and “human” factors with geekiness!

output

- A unique ID for an element on a page
- Each ID must be unique; can only be used once in the page

Linking to sections of a web page

```
<p>Visit <a href=
"http://www.textpad.com/download/index.html#downloads">
textpad.com</a> to get the TextPad editor.</p>
<p><a href="#mission">View our Mission Statement</a></p>
```

HTML

Visit textpad.com to get the TextPad editor.

View our Mission Statement

output

- Link target can include an ID at the end, preceded by a #
- Browser will load that page and scroll to element with given ID

CSS ID selectors

```
#mission {
font-style: italic;
font-family: "Garamond", "Century Gothic", serif;
}
```
CSS

Coding Horror! Coding Horror!

Our mission is to combine programming and "human" factors with geekiness!

output

- Applies style only to the paragraph that has the ID of mission

HTML class attribute

```
<p class="shout">Coding Horror! Coding Horror!</p>
<p class="special">See our special deal on Droids!</p>

<p class="special">Today only!</p>
```

HTML

Coding Horror! Coding Horror!

See our special deal on Droids!

Today only!

output

- A way to group some elements and give a style to only that group
- Unlike an id, a class can be reused as much as you like on the page

CSS class selectors

```css
.special {
background-color: yellow;
font-weight: bold;
}
p.shout {
color: red;
font-family: cursive;
}
```

CSS

Coding Horror! Coding Horror!

See our special deal on Droids!

Today only!

output

CSS class selectors

```
<p class="shout">Coding Horror! Coding Horror!</p>
<p class="special">See our special deal on Droids!</p>

<p class="special shout">Today only!</p>
```

HTML

Coding Horror! Coding Horror!

See our special deal on Droids!

Today only!

output

CSS ID selectors

```
a:link { color: #FF0000; } /* unvisited link */
a:visited { color: #00FF00; } /* visited link */
a:hover { color: #FF00FF; } /* mouse over link */
```

CSS

Buy Early Buy Often!

output

CSS ID selectors

class	description
:active	an activated or selected element
:focus	an element that has the keyboard focus
:hover	an element that has the mouse over it
:link	a link that has not been visited
:visited	a link that has already been visited
:first-letter	the first letter of text inside an element
:first-line	the first line of text inside an element
:first-child	an element that is the first one to appear inside another