

CISC 1600 Lecture 1.2

Introduction

Topics:

Markup

HTML box model

Document object model (DOM)

Cascading style sheets (CSS)

Markup Languages (defined).

From: http://en.wikipedia.org/wiki/Markup_language

“A markup language is a system for annotating a document in a way that is syntactically distinguishable from the text. The idea and terminology evolved from the "marking up" of paper manuscripts, i.e., the revision instructions by editors, traditionally written with a blue pencil on authors' manuscripts.”

Editing/Proofreading Symbols (a type of markup language)

Symbol	Meaning	Example
↵	Insert a comma	A stitch in time [↵] saves nine.
↱	Apostrophe or single quotation mark	The woodchuck couldn't chuck wood.
^	Insert something	An Apple [^] keeps the doctor away. <i>day</i>
“ ”	Use double quotation	“The better to see you with.” said the wolf.
⊙	Use a period here	Mary had a little lamb [⊙]
~	Delete	The cow jumped over the full moon.
↔	Transpose elements	Jack <u>over</u> jumped the candlestick.
)	Close up this space	The win ⁾ dows on the bus goes up and down.
#	A space is needed here	And they all lived happily ever [#] after.
¶	Begin new paragraph	“Knock, knock.” [¶] “Who’s there?”
No ¶	no paragraph	“I’ll huff and puff” said the wolf. “And I’ll blow your house down!” ^{No ¶}

Markup Languages are not the same as programming languages

- Programming languages are used to create programs that control the behavior of a machine.
 - C/++/#, Java, Processing, Python, PHP, Ruby, Haskell
- Markup languages are used for adding information (sometimes called metadata) to text in a way which is distinguishable from that text.
 - HTML, LaTeX, GenCode, SGML, XML
- It is possible to embed programming language statements / commands into a markup language.

Types of Markup

(HTML does all three)

- **Presentational markup:** Used by traditional word-processing systems, to create a WYSIWYG effect. Examples: add a line break, bold a word, change font style or color.
- **Procedural markup:** Provides instructions for programs that are to process the text. Examples: add an image, video, or link to a document.
- **Semantic markup:** Used to label parts of a document and attach additional meaning to those sections. Examples: define the title of a document or declaring that a section of text is an address.

Motivation: Plain Text

Computer and Information Science CISC 1600: Introduction to Multimedia Computing Spring, 2016 (3 hours, 3 credits) Description. Introduction to multimedia topics, including: web design, game design, animation, data visualization, simulation and robotics. Introduction to multimedia hardware and software, including game boxes. Human interface design and input using multimedia devices. Graphical and other forms of output to multimedia devices. Emphasis on design and creation of web pages with HTML and cascading style sheets; interactive, graphical web-based programs; simple computer games, movies and narratives. Computer-based sound editing. Introduction to agent-based programming for simulations and robotics. Uses of multimedia in industry. Hands-on exercises. Instructor: Prof. Michael Mandel. Email: mim@sci.brooklyn.cuny.edu Phone: 718-951-5600 x2053 Office: 2232N Web: <http://mr-pc.org> Office hours Monday 7–8 pm, Thursday 10–11 am, Thursday 12:15–1:15 pm and by appointment Course meetings Tuesday and Thursday 11:00–12:15 pm, WH-206 Prerequisites. [None] Textbook. There is no textbook for the course. Online Resources. Slides, labs, assignments, and readings will be posted on the course website: <http://mr-pc.org/t/cisc1600/> Grading. The course will be graded on a curve, with the final grade computed by combining individual assignments as follows: Participation / attendance Labs (x10) Projects (x3) Homeworks (x2) Midterm Final exam 10% 10% 30% 8% 12% 30% All homeworks and projects should be turned in at the beginning of the corresponding class period. Attending class is mandatory and attendance will be taken at the beginning of every meeting. This rule does not apply to absences due to religious observances, as described on page 72 of the Undergraduate Bulletin. Course Objectives

Text augmented with presentational markup

syllabus.pdf

m.mr-pc.org/t/cisc1600/2016sp/syllabus.pdf

Search

Scholarfy JSALT ws15FFS Portal Portal S3 English tumblr OsuLib Gmail! Mendeley

Page: 1 of 2 Automatic Zoom

Computer and Information Science
CISC 1600: Introduction to Multimedia Computing
Spring, 2016
(3 hours, 3 credits)

Description Introduction to multimedia topics, including: web design, game design, animation, data visualization, simulation and robotics. Introduction to multimedia hardware and software, including game boxes. Human interface design and input using multimedia devices. Graphical and other forms of output to multimedia devices. Emphasis on design and creation of web pages with HTML and cascading style sheets; interactive, graphical web-based programs; simple computer games, movies and narratives. Computer-based sound editing. Introduction to agent-based programming for simulations and robotics. Uses of multimedia in industry. Hands-on exercises.

Instructor: Prof. Michael Mandel

Email mim@sci.brooklyn.cuny.edu
Phone 718-951-5600 x2053
Office 2232N
Web <http://mr-pc.org>
Office hours Monday 7–8 pm, Thursday 10–11 am, Thursday 12:15–1:15 pm and by appointment

Course meetings Tuesday and Thursday 11:00–12:15 pm, WH-206

Prerequisites [None]

Textbook There is no textbook for the course.

Online Resources Slides, labs, assignments, and readings will be posted on the course website:
<http://mr-pc.org/t/cisc1600/>

Document: 100% Images: 0/0 Loaded: 28 KB Speed: 61.95 KB/s Time: 0.453

Back to HTML and CSS

Every HTML element is a box

Hello world!

This is the first paragraph

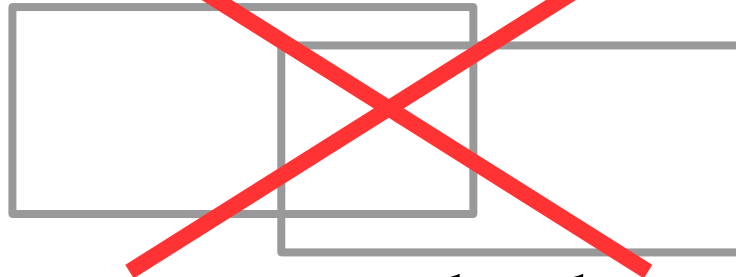
Every HTML element is a box

Hello world!

This is the first paragraph

Boxes cannot partially overlap

- Two boxes cannot partially overlap with each other



- They either have to completely overlap



- Or be completely separate



Boxes can be organized into a tree

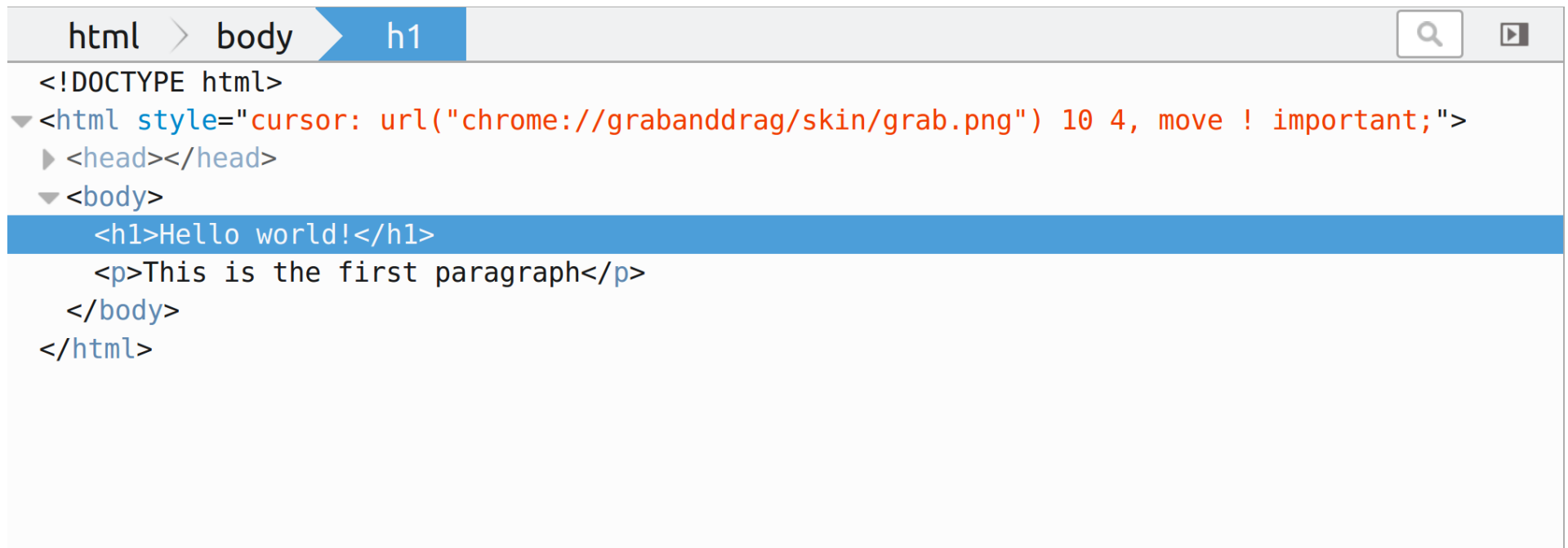
Hello world!

This is the first paragraph

Document Object Model (DOM)

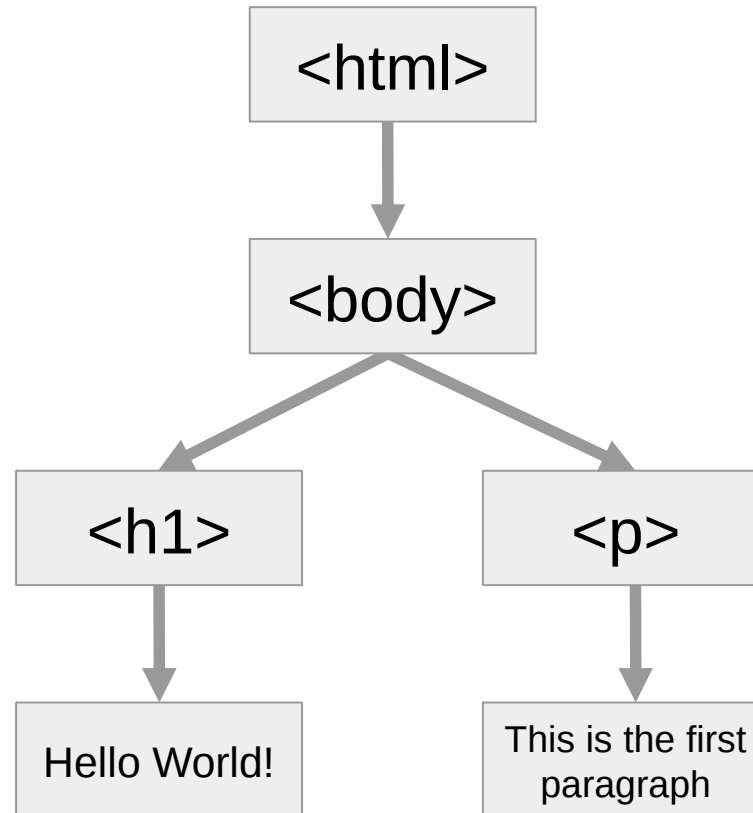
- Because the boxes (elements) can't overlap, they can be put into a hierarchy
 - The hierarchy can be represented as a “tree”
 - The DOM tree
- When building a page, consider this structure first, i.e., the DOM
- DOM tree can be manipulated by CSS and javascript after it is built

Example DOM



```
html > body > h1
<!DOCTYPE html>
<html style="cursor: url('chrome://grabanddrag/skin/grab.png') 10 4, move ! important;">
  <head></head>
  <body>
    <h1>Hello world!</h1>
    <p>This is the first paragraph</p>
  </body>
</html>
```

Example DOM



Cascading style sheets (CSS)

- CSS provides a way to style DOM elements
 - Different syntax from HTML
- “Cascading” because rules cascade down the DOM tree from root to leaf
- Controls both styling and positioning

CSS Syntax

- General form of a declaration:
`selector { property: value; property: value; }`
- Selector is usually the name of a tag
- Braces are curly { }
- Properties are separated from values by colons :
- Property-value pairs are terminated with semicolons ;
- White space is ignored

CSS example declarations

- Simple:

```
p { margin: 10pt; }
```

- Multiple properties in one block:

```
h1 {  
    font-family: Verdana, sans-serif;  
    color: red;  
    font-size: 20px;  
}
```

- Multiple selectors, relative sizing:

```
p, div, h2 { color: #00ddff ; width: 80%; }
```

There are three ways to include CSS in HTML

- Include an external CSS file
- Include CSS declarations directly in the head of your document
- Include CSS declarations in style attribute of individual elements
- Can use any combination
 - the “closest” definition is used

Include an external file

- Use a link tag in the head of your HTML document:

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

- “rel” stands for ‘RELationship’
 - type shows that it’s a text file acting as a CSS stylesheet
- You can use this tag multiple times in the same document
 - To link multiple stylesheets to the page
 - e.g., one file for fonts, another for margins and spacing

Include CSS declarations directly in the head of your document

- Put a style element into the head of your file

```
<style type="text/css">
  p { font-weight: bold; color: gray; }
  h1 { color: black; }
</style>
```

- The type attribute again tells the browser to interpret this as CSS
- Called “inline style block”

Include CSS declarations in style attribute of individual elements

- Put a style attribute in the opening tag:

```
<p style="color: blue; font-family: Ariel"> ... </p>
```

- Only applies within that element (& children)
- Note the lack of curly braces here, but the colons and semicolons are still necessary
- You shouldn't need to use this

Don't repeat yourself

- I like to be efficient (and I'm a little lazy)
- I don't want to have to solve a problem that I have solved before
- I want to **reuse** my previous solution
 - The one that I've already debugged
- The great thing about computer science is that you can do exactly that

Don't repeat yourself

When styling a website

- Cascading style sheets allow us to define styles once per website
 - And then reuse those definitions on all of its pages
- Changes to the centralized style will automatically be propagated to all pages
- The CSS file can be downloaded once by the browser and saved to speed up page loading

Separation of concerns: presentation vs content

- Content is the “what” of the document:
 - text, media (e.g., images), structure
- Presentation is the “how”
 - How should the page be displayed to the user?
- They are only loosely coupled to each other
- Different people could be responsible for each
- This is a general example of the concept of a “separation of concerns”

Classes and IDs

- How would you style different instances of the same element differently?
 - Paragraphs in the main article vs in the sidebar
- Or how would you style different non-nested tags the same?
- Use the “class” and “id” attribute for any tag

Classes

- In the HTML file, classes are attributes:

```
<p class="caution"> ... </p>
```

- In the CSS file, classes are selected by preceding them with a dot:

```
.caution { font-size: 200%; color: red; }
```

- Multiple tags in the same page can have the same class

IDs

- IDs are almost the same as classes
- But there can be at most one element in a single HTML page with a given ID
- An ID is a unique identifier of that element
- In the HTML:

```
<ul>  
    <li id="item1"> ... </li>  
    <li id="item2"> ... </li>  
</ul>
```

- In the CSS:

```
#item1 { background: gray; }
```

Span and div tags

- With classes and IDs as selectors, we could just use the same tag for everything with different classes:

```
<div class="big-red-box">  
  <span class="caution"> Warning:</span> wrong password  
</div>
```

- `<div>` acts like the boxes we saw earlier
- `` lets you apply a class/id to running text
- While possible, it's better to use new HTML semantic tags, which are standardized and give the browser more information

CSS files can be validated

- Can check validity of CSS files and blocks

<http://jigsaw.w3.org/css-validator>

- Will flag:
 - Syntax errors
 - Invalid properties
 - Invalid values
 - Etc.