中山大學
SUN YAT-SEN UNIVERSITY

Lecture 2
HTML and CSS Basics

SE-805 Web 2.0 Programming

(http://my.ss.sysu.edu.cn/wiki/display/W2PSC/Home , **supported by Google;**
using some slides of & inspired by Marty Stepp's CSE 190 M courseware)

School of Software, Sun Yat-sen University

Outline

- **Basic HTML**
- Web Standards
- Basic CSS
- Thinking…

Hypertext Markup Language (HTML)

- 1993: Initial official proposed description of HTML submitted to the IETF standards group.
- 1995: HTML 2 becomes an official standard language by a publication called RFC 1866.
- 1996-97: HTML 3.2 standardizes various features including forms, tables, image maps, and internationalization.
- 1997: HTML 4 is proposed by W3C standard body, adding style sheets, scripting, frames, embedding objects, internationalization, and accessibility for disabilities.
- 1999: HTML 4.01 the last major version of the language is published by W3C. **A majority of pages on the Web today still use it as their started language**.
- 2001-01: XHTML, HTML based on XML

Now and future, **HTML5**!

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 contents with opening and closing **tags**
- each tag's name is called an **element**
  - syntax: <*element*> *content* </*element*>
  - example: This is an paragraph
- most whitespace is insignificant in HTML (ignored or collapsed to a single space)
- we will use HTML5

Hypertext Markup Language (HTML)

- HTML coding convention: the structure of HTML is a tree.
  - indent nested elements
  - separate siblings with blank line when it makes reading easy.

- **Responsibility of HTML languages**
  - HTML describes the content and structure
  - Style Sheets (CSS) describes the appearance of the document
  - Script (JavaScript) describes the behavior of the document

- index.html
  - http://www.sysu.edu.cn/ = http://www.sysu.edu.cn/index.html

Comments: <!-- ... -->

comments to document your HTML file or “comment out” text

```
<!-- My web page, by Tim Student SS 12345, Spring 2048 -->
<p>SS courses are <!-- NOT --> a lot of fun!</p>
```

HTML

SS courses are a lot of fun!

output

- comments are still useful for disabling sections a page
- comments cannot be nested and cannot contain a –
- many web pages are not thoroughly commented (or at all)
  - comment is a communicative approach, to explain your designs and purposes to your colleagues, or even yourself sometime later.
  - comment is not for browsers of end users, but for developers and designers.

Structure of HTML page

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html> PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html> xmlns="http://www.w3.org/1999/xhtml">
    <head>
        information about the page
    </head>
    <body>
        page contents
    </body>
</html>
```

HTML5

- the header describes the page and the body contains the page's contents
- an HTML page is saved into a file ending with extension .html

Page title: <title>

describes the title of the Web page

```
<title>Chapter 2: HTML Basics</title>
```
HTML

- placed within the **head** of the page
- displayed in the Web browser's title bar and when bookmarking the page

Page meta data: <meta>

describe meta data of the Web page

```
<meta name="description" content="introduction of SYSU" />
```

HTML

```
<meta charset="gbk" />
 content="text/html; charset=gbk" />
```

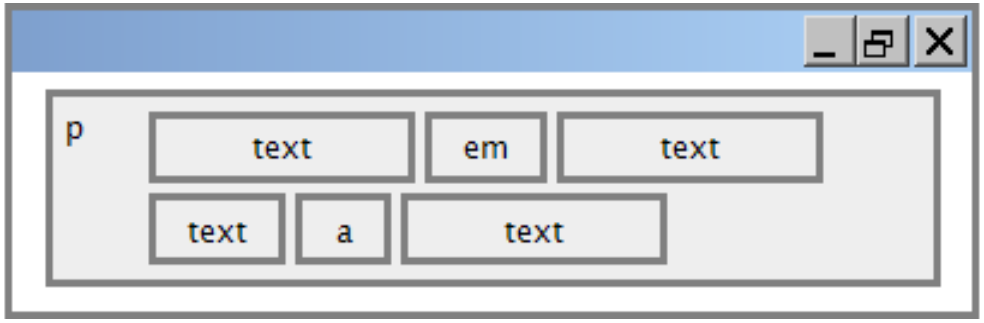
HTML5

- placed within the **head** of the page
- **chartset** is very significant in practice, and we often use **utf-8** for language other than English
  - character encoding and decoding, where, when, and how?

Character Encoding

- manipulating and representing of characters in computer
  - the bit length of the character
  - and the proper visual symbol
  - encoding vs. decoding
- chartset
  - ASCII(basic 7b, extension 1B),
  - **iso-8859-1/latin-1** (West Europe,1B）
  - **GB2312** (2B, Simplified Chinese)
  - **GBK**(2B, S. & T. Chinese)
  - **BIG5** (2B, Traditional Chinese)
  - GB18030 (1,2,4B, Eastern Asia)
  - Unicode (650 languages)
    - **UTF-8** (1,2,3,4B , Chinese 3B)
    - **UTF-16** (2B, 4B, Chinese 2B)
    - UTF-32 (4B, future)
  - UCS
    - UCS-2 (2B, comparable with UTF-16)
    - UCS-4 (4B, future)

Block and inline elements



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

Paragraph: <p>

```
<p>You're not your job. You're not how much money you have
in the bank. You're not the car you drive. You're not the
contents of your wallet. You're not your khakis. You're
the all-singing, all-dancing crap of the world.</p>
```

HTML

You're not your job. You're not how much money you have in the bank. You're not the car you drive. You're not the contents of your wallet. You're not your khakis. You're the all-singing, all-dancing crap of the world.

output

- placed within the **body** of the page
- more paragraph examples

Line break:

forces a line break in the middle of a block element (*inline*)

```
<p>Teddy said it was a hat, <br /> So I put it
on.</p> <p>Now Daddy's sayin', <br /> Where the
heck's the toilet plunger gone?</p>
```
HTML

Teddy said it was a hat,
So I put it on.
Now Daddy's sayin',
Where the heck's the toilet plunger gone?

output

- **br** should be immediately closed with **/>**

Headings: <h1>, <h2>, …<h6>

headings to separate major areas of the page (*block*)

```
<h1>Sun Yat-Sen University</h1>
<h2>School of Software</h2>
<h3>Support by Google</h3>
```

HTML

Sun Yat-Sen University

School of Software

Support by Google

output

- more heading examples

<article>

源代码 2-8 article 元素示例

```
<h1>会当凌绝顶</h1>
<article>
  <h2>五岳</h2>
  <h3>泰山：会当凌绝顶，一览众山小</h3>
  <p>  泰山……</p>
  ……
</article>
<article>
  <h2>四大佛教名山</h2>
  <h3>峨眉山：峨眉天下秀</h3>
  <p> 峨眉山……</p>
</article>
```

- <article> is for semantic, not for content

Semantic HTML

- If you find the following code is shown too big in your browser, what will you do?

```
<h1>Sun Yat-Sen University</h1>
```
HTML

Sun Yat-Sen University

output

- make it from **h1** to **h3**?
- Semantic HTML – **Separation of concerns**
  - choosing tags based on the meaning of the content rather than its appearance
  - flexible and reusable

Horizontal rule: <hr />

headings to separate major areas of the page (*block*)

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

HTML

First paragraph

Second paragraph

output

- should immediately closed with />

More about HTML tags

- some tags can contain additional information called **attributes**
  - syntax:
    ```
    <element attribute1="value1" attribut2="vaule2">
    content</element>
    ```
  - example: `<a href="page2.html">Next page</a>`
- some tags don't contain content; can be opened and closed in one tag
  - syntax: `<element attribute1="v1" attribut2="v2" />`
  - example: `<hr />`
  - example: `<img src="bunny.jpg" alt="bunny" />`

Links: <a>

links, or “anchors”, to other pages (*inline*)

```
<p>
  Search
  <a href="http://www.google.com/">Google</a> or
our
  <a href="lectures.html">Lecture Notes</a>
</p>
```

HTML

Search Google or our Lecture Notes.

output

- uses the **href** attribute to specify the destination URL
  - can be **absolute** (to another Web site) or **relative** (to another page on this site)
- anchors are inline elements; must be placed in a block element such as p or h1

Links: <a>

- hover a link in a browser, its destination URL will be shown on the status bar
- **be descriptiveness!**

Click here to check your course schedule

output

Please check your course schedule

output

Course Schedule (please check yours before March 15!)”

output

- What’s principle applied here?
- **Kind**.
  - you are kind to your Web page readers by making the page descriptive, which in turn let them understand easier.

Images: <img>

inserts a graphical image into the page (*inline*)

```
<img src="imges/gollum.jpg" alt="Gollum form LOTR" />
```

HTML

output

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

More about images

```
<a href="http://theonering.net/">
    <img src="images/gandalf.jpg" alt="Gandalf"
      title="You shall not pass!" />
</a>
```

HTML

output

- if placed inside an anchor, the image will become a link
- the **title** attribute specifies an optional tooltip
- `images/gandalf.jpg` vs. `/images/gandalf.jpg`

Phrase elements: ,

em: emphasized text (usually rendered in *italic*)

strong: strongly emphasized text (usually rendered in **bold**)

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

HTML

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

output

- as usual, the tags must be properly nested for a valid page
- em vs. i, strong vs. b
  - SoC again~!

Nesting tags

Bad:

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

- 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

Unordered list: <ul>,

ul represents a bulleted list of items (*block*)

li represents a single item within the list (*block*)

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

HTML

- No shoes
- No shirt
- No problem!

output

More about unordered lists

a list can contain other lists:

```
<ul>
  <li>Simpsons:
    <ul>
      <li>Homer</li>
      <li>Marge</li>
    </ul>
  </li>
  <li>Family Guy:
    <ul>
      <li>Peter</li>
      <li>Lois</li>
    </ul>
  </li>
</ul>
```

HTML

- Simpsons:
  - Homer
  - Marge
- Family Guy:
  - Peter
  - Lois

output

Ordered list:

ol represents a numbered list of items (*block*)

```
<p>RIAA business model:</p>
<ol>
  <li>Sue customers for copying music</li>
  <li>???</li>
  <li>Profit!</li>
</ol>
```

HTML

RIAA business model:

1. Sue customers for copying music
2. ???
3. Profit!

output

Outline

- Basic HTML
- **Web Standards**
- Basic CSS
- Thinking…

Web standards

- It is important to write proper XHTML code and follow proper syntax.
- 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.

XHTML versions

- *XHTML 1.0* (W3C Recommendation)
  - HTML 4.01 with XML syntax
  - XHTML 1.0 Strict, XHTML 1.0 Transitional, XHTML 1.0 Frameset
- *XHTML 1.1* (W3C Recommendation)
  - module-based XHTML
  - Ruby characters 北 京 (ㄅㄟˇ ㄐㄧㄥ) (běi jīng)
- *XHTML 1.2*
  - improved Semantic Web support through RDFa
  - draft, not widely adopted
- *XHTML 2.0*
  - not backward compatible
  - only in draft, not standard, and will be expired at the end of 2009
- **HTML 5**
  - Loser's Counterattack! XHTML2.0 vs. HTML5 W3C vs. WHARWG

Outline

- Basic HTML
- Web Standards
- **Basic CSS**
- Thinking…

The bad way to produce styles

```
<p>
  <font face="Arial">Welcome to Greasy Joe's.</font>
     You will <b>never</b>, <i>ever</i>, <u>EVER</u>
     beat <font size="+4" color="red">OUR</font> prices!
</p>
```

HTML

Welcome to Greasy Joe's. You will **never**, *ever*, EVER beat OUR prices!

output

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

Cascading Style Sheets (CSS): <link>

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

Embedded in HTML

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

standalone CSS file

- **CSS** describes the appearance and layout
  - as opposed to HTML, which describes content of the page
  - either on screen and on print
- can be embedded in HTML or placed into separate .css file (preferred)

Basic CSS rule syntax

```
selector {
  property 1: value 1;
…
  property n: value n;
}
```
CSS

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

- comments : /* …. */
- a CSS file consists of one or more **rules**
- each rule starts with a **selector** that specifies an HTML element(s) and then applies style **properties** to them
  - a selector of * selects all elements

CSS properties for colors

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

CSS

This paragraph uses 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.

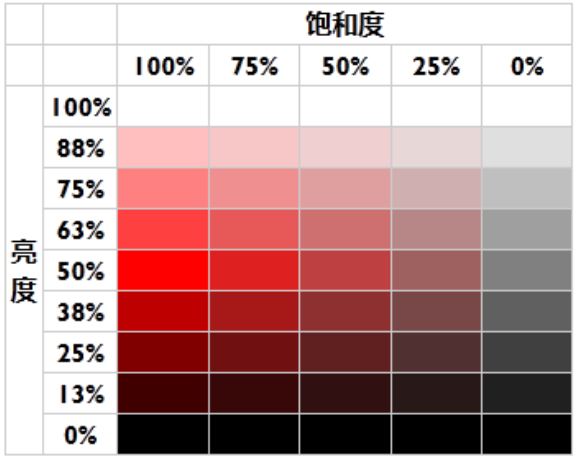
output

color names: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, (white), 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)

HTML5 Color

- RGB
- RGBA: Alpha for opacity, no hexadecimal form
  - rgba(255, 165, 0, 0.5)、rgba(100%, 65%, 0%, 0.5)
- HSL (Hue Saturation Lightness):
  - hsl（120, 100%, 50%）
- HSLA
  - hsl（120, 100%, 50%, 0.5）



HTML5 Color

- HSL is more natural for human than RGB

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	

font-family

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

This paragraph uses the Georgia font.

This h2 uses the Courier New font.
output

- Font of Chinese Characters
  - most of browsers only support SimSon（“宋体”）
  - IE can support other fonts Windows OS supported
    - 黑体：SimHei 新宋体：NSimSun 仿宋：FangSong SimFang?
      楷体：KaiTi SimKai? 仿宋_GB2312：FangSong_GB2312
      楷体_GB2312：KaiTi_GB2312 微软雅黑体：Microsoft YaHei
    - …

More about font-family

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

If no Garamond then uses TNR, and then uses serif.

output

- enclose multi-word font names in quotes
- can specify multiple fonts from highest to lowest priority
- **generic font names**:
  - serif, sans-serif, monospace, cursive

font-size

```css
p {
  font-size: 20pt;
 }
```

CSS

This paragraph uses font size 20pt.

output

- **units**: pixels (px) vs. point(pt) vs. m-size(em)
  - 16px, 16pt, 1.16em
- **vague font size**: xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger
- **percentage font sizes**, e.g.: 90%, 120%

font-weight, font-style

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

CSS

This paragraph is bold and italic.

output

- either of the above can be set to normal to turn them off

@font-face

- 源代码 3-25 @font-face 规则示例

```
<style>
@font-face {font-family:myFont;  src: url('font/wtcc02.ttf')}
</style>

td {font-family: myFont;……}
<td>酷儷海報 </td>
<td> 没有对应字体 </td>
```

图 3-23 源代码 3-25 运行效果

Outline

- Basic HTML
- Web Standards
- Basic CSS
- **Thinking…**

Thinking ...

- What's the difference?

```c
void swamp(int a, int b)
{
  int temp;
  temp = a;
  a = b;
  b = temp;
}
```
C

```html
<html>
   ….
<body>
  <h1>Supper Man</h1>
  <p>
    The guy teaches you Web.
  </p>
 ….
```
HTML

```php
<?php
  $file="1.txt";
  $fp=fopen($file,"r");
  $content= fread(
    $fp,filesize($file));
  fclose($fp);
?>
```
PHP

```css
body {
  background-color: #997788;
  font-family: SimSon;
}
h1 {
  color: blue
}
```
CSS

Declarative Programming

- Imperative vs. Declarative
  - **declarative programming** is a programming paradigm that expresses the logic of a computation without describing its control flow.
    -- Lloyd, J.W., *Practical Advantages of Declarative Programming*
  - in contrast with imperative programming, which requires an explicitly provided algorithm.
- Subparadigms
  - functional programming: Scheme, Erlang, Haskell, …
  - logic programming: Prolog
  - domain-specific languages: SQL, CSS, HTML, XSLT, SVG, XAML, regular expressions
  - constraint programming: often used as a complement to other paradigms
  - Hybrid languages: Makefiles, yacc

Summary

- HTML
  - HTML & XHTML
  - HTML Tags: title, meta, p, h1, ..., hr, a, img, br, comments, em, strong, ul, ol, li
  - block vs. inline
  - character encoding
- CSS
  - why? how?
  - link, rule
  - properties: for color, for fonts
- Thinking:
  - SoC
  - Declarative Programming

Exercises

- write a web page for yourself which contains your self-introduction, recent photos, courses selected this semester, and favorite movies
  - your introduction should be more than one paragraph
  - your courses should be a ordered list
  - your favorite movies should be a unordered list
  - having a link to the course site of SE-805
  - applying different fonts for the readability

Further Readings

- http://en.wikipedia.org/wiki/XHTML
- http://en.wikipedia.org/wiki/Cascading_Style_Sheets
- Chapter 1~8, *Web Programming with HTML, XHTML, and CSS* http://my.ss.sysu.edu.cn:8080/display/W2PSC/References+and+Books
- List of all HTML tags: http://www.w3schools.com/tags/default.asp
- List of HTML character entites: http://www.w3schools.com/tags/ref_entities.asp
- XHTML 1.1 Spec. http://www.w3.org/TR/xhtml11/
- XHTML 1.1 Elements Reference: http://www.w3.org/2007/07/xhtml-basic-ref.html
- W3 List of all CSS properties: http://www.w3.org/TR/CSS21/propidx.html
- W3 CSS 2.1 Specifications: http://www.w3.org/TR/CSS21/
- Fonts of each operating systems: http://www.apaddedcell.com/web-fonts

Thank you!


unintended uses
user contributions
effortless scalability
trust
tagging
beta
radical decentralization
customer self-service
mass service of micromarkets
software as a service
right to remix
participation
blogs
wikis
VOIP
podcasting
filtering
sharing
social search
collaboration
social bookmarking
RIAs
Ajax
SSE
RoR
BitTorrent
KatrinaList
Wikipedia
Skype
Hard to Recreate Data
Enrichment & Annotation
Copyrighted & IP Content
Unique Formulas
Software That Gets Better The More People Use it
Face
HTML, Ajax, Flash, etc.
OpenSim
Services (API)
RSS, REST, SOAP, JSON etc.
user control
small pieces
permalinks
http
two-way interaction
tcp stream
people
World-Wide Web
data
mash-up
Web 2.0