Midterm ch.3-5

> Some basic concepts may ignored.

3. HTML(Hypertext Markup Language)

- Browsers Are Tolerant

  They ignore markup they don't understand, eg. Internet Explorer/Firefox

- HTML Elements

  Some elements do not require end tags, eg. `<p>` paragraph tag

  Some elements do not require content, eg. `<hr>`horizontal rule tag

- Attributes

  Names are **case insensitive**, but not necessarily attribute values

  Some elements are deprecated, eg. background.. in element

- Comments, `<!--`, `-->`
- Composing HTML
- Lists
  - unordered `<ul></ul>`, `<li>`
  - ordered `<ol></ol>` , `<li>`, `start=4, type=A/I/…`
  - difinition `<dl></dl>`, `<dt>`, `<dd>`
  - `<menu>`
- Table
  - `<caption></caption>` to label a table, `<th></th>` header, blodface, `<td></td>` data, `<tr>` row
  - `rowspan=4` with 4 rows, `colspan=2`
  - `align right`..., but these are deprecated in HTML5
- Character Set

  Universal Character Set (UCS), eg. ISO-8859-1, ISO-8859-5, SHIFT_JIS, EUC-JP

- Character references
  - Numeric character references (either decimal or hexadecimal)

    `&#229;` : å; `&#60;` : <, left angle bracket; `&#62;` : >, right angle bracket; `&#38;` : &, ampersand sign; `&#34;` : ", double quote

  - Character entity references.

    `<` : < sign; `>` : > sign; `&` : & sign; `&quot` : " mark

- Anchors Away

  `<a href=".../mailto:/ftp:/new:/"></a>`

  - Names

Uniqueness, String matching (case-sensitive)

The id and name attributes share the same name space. They cannot both define an anchor with the same name in the same document. [name HTML5中已经废弃]

- Titles `title=""`

- Universal Resource Identifier(URI)
  - `absolute-URI = scheme ":" hier-part [ "?" query ]`
    1. The scheme of the mechanism used to access the resource.
    2. The name of the machine hosting the resource.
    3. The name of the resource itself, given as a path

- Fragment identifiers are URIs that refer to a location

within a resource. `#`

- `<LINK>`
  - only appear in the HEAD
  - To provide information to search engines. eg. Links to style sheets and “media queries” used in Responsive Web Design
- Digital cameras & Smartphones, typical: 1280x720, 1920x1080
- Image Formats
  1. x-pixelmaps(obsolete)
  2. Graphic Interchange Format (GIF, obsolete)
  3. Joint Photographic Experts Group (JPEG). Includes image compression
  4. Portable Network Graphics (PNG). An open, extensible image format with Lossless Compression
- `<img src="">`
  - `alt="Alt to replace an image with text, if the image is unavailable or a text browser is used"`
  - Active Images: `<a ...><img ... ></a>`
  - Image Maps
    1. active images with multiple clickable regions, eg

```
<img src="image.png" alt="Website map" usemap="#mapname" />
<map name="mapname">
  <area shape="rect" coords="9,372,66,397"
href="http://en.wikipedia.org/" alt="Wikipedia" title="Wikipedia"
>
</map>
```

    2. SHAPE: default(Specifies the entire region), rect, circle, poly

- `<META>`
  - Allows you to insert Name/Value pairs describing document properties.

    - eg. `name="" content=""`, `<META HTTP-EQUIV="REFRESH" CONTENT="5; URL=http://www.usc.edu/dept/cs/">`
    - Robot Exclusion, `<meta name="robots" content="noindex,nofollow">`
        - [NO]INDEX. The INDEX directive specifies if an indexing robot should index the page.
        - [NO]FOLLOW The FOLLOW directive specifies if a robot is to follow links on the page.
        - The defaults are INDEX and FOLLOW. The values ALL and NONE set all directives on or off: ALL=INDEX,FOLLOW and NONE=NOINDEX,NOFOLLOW.
        - Note the "robots" name of the tag and the content are case insensitive.
- Validating Your HTML
    - Reason: Browsers display & treat HTML differently
    - Validators, eg:
        1. Flag syntax errors with respect to HTML DTD
        2. Compare your pages to HTML 4.x, XHTML, and even HTML 5 (experimental)
- `<EMBED>`

  `<APPLET>` allows designers to embed a Java applet in an HTML document.(deprecated)

  eg. `<embed type="video/quicktime" src="movie.mov" width="640" height="480">`
- HTML Tidy: W3C Markup Validator Service

4. HTML: Style Sheets

- `<STYLE type="text/css"></STYLE>`, or `<LINK href="special.css" rel="stylesheet" type="text/css">`, or Inline Style Attribute
- Color: #f00, #ff0000, rgb(255,0,0), rgb(100%,0%,0%), pre-defined name
- Selectors: `.`, `#` used once (case sensitive); at specific tags
- Inheriting STYLE Properties
    - and tags have no initial presentation properties
        - exception, line break before and after a tag
        - applies to **inline** elements (example: **)**
        - **applies to block elements (example:**

)
    - **With CSS, properties such as text-align are inherited from the parent element**
- **Precedence of Style Settings**
    - **`style attribute` > `<style>` > `file.css`**
    - **`tag.class` > `.class` > `tag`**
- **`!DOCTYPE`**

Instructs modern browsers to work in 'standards compliant mode', or browsers work in ‘Quirks’ mode

- Internet Explorer will display fonts larger than standards mode. e.g. fonts rendered in 12pt (medium) rather than 10pt
- IE Uses the ‘broken box model’. eg. Measures the dimensions of a box using the inner size, not the outer size as in standard mode
- Loose compliance (minimum requirement): `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">`
- Strict compliance: `<!DOCTYPE html PUBLIC “@-//W3C//DTD XHTML 1.0 Strict//EN” “http://www.w3.org/TR/html4/strict.dtd”>`
- XHTML Transitional compliance (less strict): `<!DOCTYPE html PUBLIC “@-//W3C//DTD XHTML 1.0 Transitional//EN” “http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional-dtd”>`
- HTML5: `<!DOCTYPE html>`

- Media Types,

  create documents for different media types

  eg.

```
<STYLE type=text/css media=projection>
H1 {color:blue}
</STYLE>

CSS3
<!--Target specific physical characteristics of device.-->
<link rel="stylesheet" type="text/css" media="screen
and (max-device- width: 480px)" href="min.css" />

<!--two equivalent pair of media queries-->
<style>
...
@media all and (min-width:500px) { ... }
@media (min-width:500px) { ... }
</style>
```

- Pseudo Elements and classes
  - pseudo-classes
    1. :link – a normal, un-visited link
    2. :visited – a link the user has visited
    3. :hover - a link when the user mouses over it
    4. :active - a link the moment it is clicked
  - pseudo classes

    :first-child; :hover; :active, selects the active link; :focus, selects the input element which has the focus; :lang, selects every

element with a lang attribute

- pseudo elements

  :first-line; :first-letter; :before; :after, to insert some content after the content of an element

- Box Model: Margin(transparent) - Border - Padding - Content

  eg. Collapsed margin-top is max(30px, 12px) = 12px;

```
<HTML><TITLE>Examples of margins, padding, and borders</TITLE>
<STYLE type="text/css">
UL { background: green;
margin: 12px 12px 12px 12px;
padding: 30px 3px 3px 3px; /* No borders set */ }
LI { color: black; /* text color is black */
background: gray; /* Content, padding will be gray */
margin: 12px 12px 12px 12px;
padding: 12px 0px 12px 12px; /* Note 0px padding right */
list-style: none /* no glyphs before a list item */
/* No borders set */ }
LI.withborder { border-style: dashed;
border-width: medium; /* sets border width on all sides */
border-color: black; } </STYLE> </HEAD>
<BODY> <UL> <LI>First element of listFirst element of listFirst element of listFirst element of a a a a list
<LI class="withborder">Second element of list is longer to illustrate wrapping.First element of a a a a a listFirst element of listFirst element of list
</UL>
</BODY>
</HTML>
```

- CSS Vendor Prefixes

  1. Android/Chrome/iOS/Safari: -webkit-
  2. Firefox: -moz-
  3. Internet Explorer: -ms-
  4. Opera: -o-

  eg.

```
/*before HTML 5, to set a rounded corner on a box one would have to write*/
-moz-border-radius: 10px 5px
-webkit-border-top-left-radius: 10px;
-webkit-border-top-right-radius: 5px;
-webkit-border-bottom-right-radius: 10px;
-webkit-border-bottom-left-radius: 5px;
border-radius: 10px 5px;
```

- **Reset CSS: a short, often compressed (minified) set of CSS rules that resets the styling of all HTML elements to a consistent baseline.**
- **CSS3 New Features**
  - **“modern browsers”: IE9+, Edge, Opera 10+, Firefox 3.5+, Chrome, Safari 3+**
  - **eg. box-sizing, border-radius, box-shadow, RGBA Colors, HSLA Colors, Multiple Backgrounds, background-clip, background-origin, background-size, Transforms, Media Queries. [Above supported]**
  - **eg.** `border-image: url(border.png) 30 30 round/stretch;` **, round: tiled (repeated) to fill the area; stretch: stretched to fill the area. [IE not support]**
  - **eg. Multi-Column Layout.** `column-count: 3; column-gap: 40px; column-rule: 4px outset #ff00ff;` **[IE 9 and earlier not support]**

JavaScript Basics

- **What**
  - **a "simple", interpreted, programming language with elementary objectoriented capabilities**
  - **Basics**
    1. **case-sensitive**
    2. **ignores spaces, tabs, newlines -> minified**
    3. **Semicolon is optional**
    4. **C and C++ style comments are supported**
  - 1. **client-side JavaScript runs on Web browsers**

       Limitations:

       1. **[Was] Extremely difficult to explicitly draw graphics.**
       2. **No access to the underlying file system or operating system**
       3. **Unable to open and use arbitrary network connections**
       4. **No support for multithreading**
       5. **[Was] Not suitable for computationally intensive applications**
    2. **server-side JavaScript runs on Web servers**
  - **designed for manipulating web pages**
    1. **Control Web page appearance and content (this is its intended use)**
    2. **Control the Web browser, open windows, test for browser properties**
    3. **Interact with document content**
    4. **Retrieve and manipulate all hyperlinks**
    5. **Interact with the user, sensing mouse clicks, mouse moves, keyboard actions**
    6. **Read/write client state with cookies**
- **Embedded**

```
<BODY>
</BODY><SCRIPT LANGUAGE="JavaScript">
//the Javascript here produces content for the BODY on
loading
</SCRIPT>
</BODY>

<HEAD>
<SCRIPT LANGUAGE="JavaScript">
//the Javascript here creates functions for later use
</SCRIPT>
</HEAD>
```

- **Events**
  - **Mouse Events: onclick, ondblclick(user double-clicks an element), onmouseover, onmouseout**
  - **Keyboard Events: onkeydown, onkeyup**
  - **Object Events: onload(browser has finished loading the page), onunload, onresize(a document view is resized), onscroll(a document view is scrolled)**
- **Literals**
  - **numbers**
    - **treated as floating point**
    - **Octal (begin with a zero), 01234**
    - **Hexadecimal (begin with zero and x), 0xFF**
  - **boolean: true, false, (also null and undefined)**
  - **string(within single or double quotes)**
    - **immutable, cannot modify after instantiation**
    - **call methods, it will not change the initial string, but return the modified string**
    - **eg. string.substring(indexA, indexB) -> [A,B)**
- **Escape Notation**

  \b backspace,\f form feed \□ single quote, \n newline, \r carriage return, \t tab, \' single quote, \" double quote, `\\` backslash
- **Reserved Words**
  1. **JavaScript identifiers start with a letter, $, or underscore followed by zero or more letters or digits;**
  2. **JavaScript reserved words: abstract, arguments, boolean, break...**
  3. **avoid using the name of JavaScript built-in objects, properties, and methods: Array, Date, eval, function, hasOwnProperty, NaN("Not-a-Number" value)...**
- **Variables**
  1. **declaration, initialization, [without type]**
  2. **The type of value a variable can hold during execution may change.**

3. **Scope: local/global**

   In a multi-frame or multi-window set up of the browser, scripts can access global variables from any other document currently loaded

- **Data Types: String, Number, Bollean, Null, Object, Function**
- **Array literal**
  1. **a list of zero or more expressions, []**
  2. **elements can have different types**
  3. **length = largest integer property name in the array + 1**
  4. **NOT provide a way to declare the size (dimension) of an array**

```
Array.dim = function (dimension, initial) {
    var a = [], i;
    for (i = 0; i < dimension; i += 1) {
        a[i] = initial; }
    return a;
};

var myArray = Array.dim(10,0); //makes an array of ten zeros
```

  5. **Iterate**

```
for (i=0; i < len; i++) {. . . }
for (x in person) { . . . }
while (condition) {. . . }
```

  6. **Arrays and Objects are Semantically Identical,** `typeof(array) = typeof(object) = object`
  7. **NOT support associative arrays**

```
var person = new Array();
person["firstname"] = "John";
person["age"] = 41;
//person["firstName"] returns "John"
//person[0] returns undefined
//person.length returns 0
```

  8. **JavaScript properties that begin with a digit cannot be referenced with dot notation; and must be accessed using bracket notation.**

```
console.log(arr.0); // a syntax error, because the property name is not valid:
```

- **Objects**

1. a list of zero or more pairs of property names and associated values of an object, {}.

   can be nested within objects

```
var myHonda = {color: “red”,
    wheels: 4,
    engine:{cylinders: 4,
            size: 2.2}
};
```

2. the "dot" operator is used to access the value of an object's property or to assign it a value

```
lname = person.lastName // returns “Doe”
```

3. Constructors, `new`

```
function cat(name, meow) {
  this.name = name;
  this.talk = function() {
    alert(this.name + " say " + meow)
  }
}
cat1 = new cat("felix", "purr");
cat1.talk();
cat2 = new cat("ginger", "hisss");
cat2.talk();
```

4. Predefined JS Objects
   - Array Object
   - Boolean Object

     `var booleanObject = new Boolean(value);`

     not (undefined, null, 0, Nan or the empty string, including a Boolean object whose value is false), evaluates to true
   - Date Object, `var Xmas95 = new Date(“December 25, 1995”);`, `Xmas95.getMonth()` returns 12
   - Function object, `Var functionObjectName = new Function([arg1, ..., argn], functionbody);`
   - Math object: includes properties and methods for mathematical constants, e.g. sin(), cos(), ceil(), floor()
   - RegExp object
   - String object

- Popup Boxes: alert(), confirm(), prompt()

- **Common Mistakes**
  1. **Undefined may not be null**

```
null == 0; // false
undefined == ""; // false
null == false; // false
undefined == false; // false

null == undefined; // true
null !== undefined; // true
null === undefined; // false
```

  2. **cannot overload a function**

     JavaScript will simply use the latest-defined version of the function and call it;

     If a parameter is omitted it is undefined
  3. **Undeclared variables are global**

     If a variable is NOT declared using var, then it is global.

     Two variables of the same name, both undeclared will create

     conflicts that are hard to debug
- **ECMAScript**
  - **ECMA(European Computer Manufacturers Association)**
  - **Current language specification is ECMA-262, 8th Edition(June 2017)**