CS2204 Fundamentals of Internet Applications Development

Lecture 3 HTML – Part 2

Computer Science, City University of Hong Kong

Semester A 2023-24

Outline

Lists

Table

Form

Multimedia

HTML: Lists

- A **list** enhances the presentation of information under a category with many items
  - E.g., which of the following webpage has better readability to you?

Web Development

Often you will find 3 components in a webpage source code:

1. HTML: HyperText Markup Language (HTML) uses a set of codes called tags to describe the structure of a webpage
2. CSS: Cascading Style Sheets (CSS) describes how the HTML elements should be displayed by specifying the fonts, colors, layout and placement of these HTML elements
3. Javascript: Javascript is a programming language that can provide instructions for a browser to dynamically generate content for a website or enhance the website interactivity

Web Development

Often you will find 3 components in a webpage source code:

1) HTML: HyperText Markup Language (HTML) uses a set of codes called tags to describe the structure of a webpage; 2) CSS: Cascading Style Sheets (CSS) describes how the HTML elements should be displayed by specifying the fonts, colors, layout and placement of these HTML elements; 3) Javascript: Javascript is a programming language that can provide instructions for a browser to dynamically generate content for a website or enhance the website interactivity

HTML: Lists (2)

- An **unordered** list displays items with bullets (by default)

 and define the beginning and end of an unordered list
 and enclose each list item

```
<h4>An Unordered List:</h4>
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

HTML

An Unordered List:

- Coffee
- Tea
- Milk

Browser display

- An **ordered** list displays items with automatic numbering

 and define the beginning and end of an ordered list
 and enclose each list item

```
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

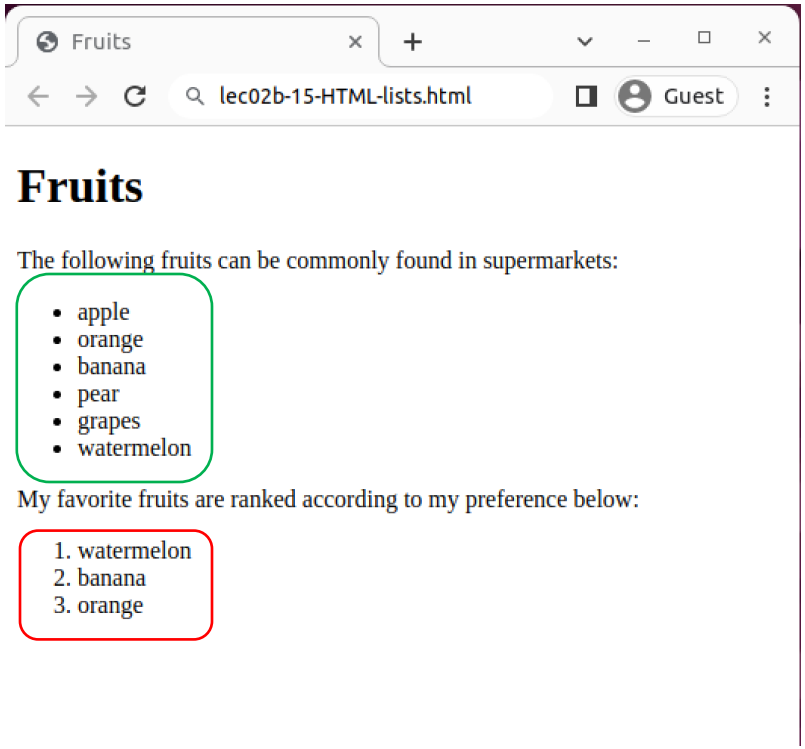
HTML

1. Coffee
2. Tea
3. Milk

Browser display

HTML: Lists Example

```html
<!DOCTYPE html>
<html>
  <head>
    <title>Fruits</title>
  </head>
  <body>
  <!-- Page content begins here -->
    <h1>Fruits</h1>
    <p>The following fruits can be commonly found in supermarkets:</p>
    <ul>
        <li>apple</li>
        <li>orange</li>
        <li>banana</li>
        <li>pear</li>
        <li>grapes</li>
        <li>watermelon</li>
    </ul>

    <p>My favorite fruits are ranked according to my preference below:</p>
    <ol>
        <li>watermelon</li>
        <li>banana</li>
        <li>orange</li>
    </ol>
  <!-- Page content ends here -->
  </body>
</html>
```



Code Example: lec02b-15-HTML-lists.html

With lists, it would be easy to insert or delete items, or rearrange the order of the items. For numbered list, there is no need to worry about the numbering even when the items are modified because the numbering is automatic

List: attributes

Sometimes an attribute can be defined in the start tag with the format `name="value"` to provide additional information

```
<ol type="A">
    <li>apple</li>
    <li>orange</li>
    <li>banana</li>
    <li>pear</li>
    <li>grapes</li>
    <li>watermelon</li>
</ol>
```

A. apple
B. orange
C. banana
D. pear
E. grapes
F. watermelon

The attribute `type` can be set with value `"A"` for an ordered list so that the number style will be A,B,C,... instead of the default 1,2,3,...

```
<ol start="3">
    <li>orange</li>
    <li>apple</li>
    <li>grape</li>
</ol>
```

3. orange
4. apple
5. grape

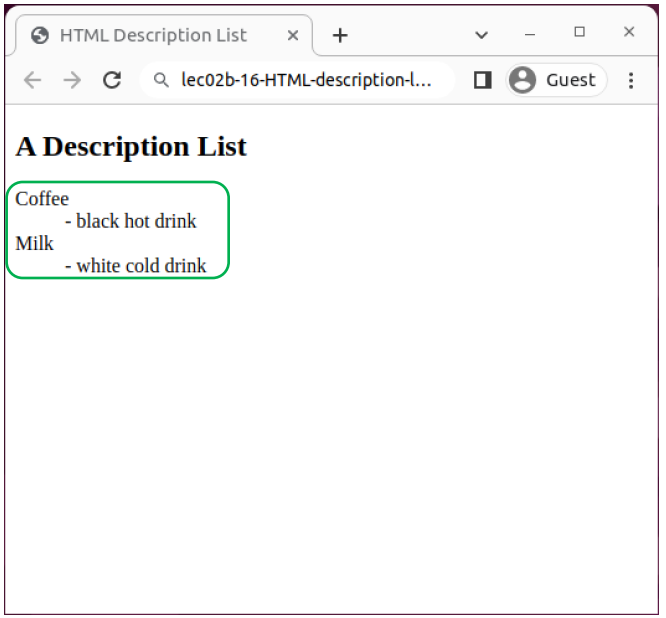
The attribute `start` can be set with a numeric value for an ordered list that corresponds to the start number of the first item instead of the default value of 1

HTML: Definition List (Description List)

- A description list is a list of terms, with a description of each term
  - `dl` for **list**, `dt` for **title** and `dd` for **data**
  - Commonly used in repeating groups of heading and description (title-data)

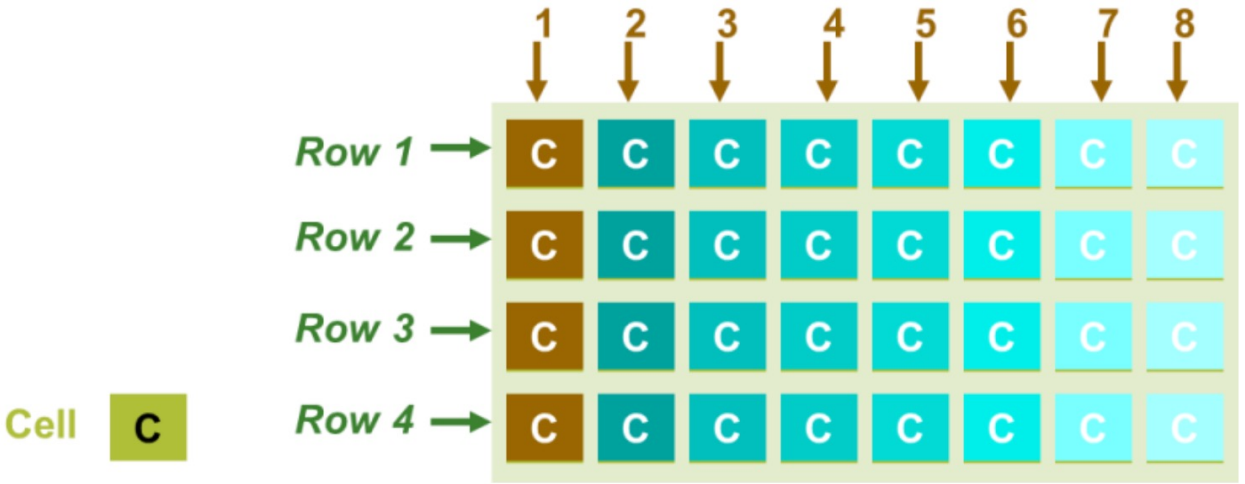
```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="author" content="CS2204 Instructor" />
    <meta name="description" content="CS2204 - HTML Lists">
    <meta name="keywords" content="CS2204 HTML, Lists" />
    <title>Fruits</title>
  </head>
  <body>
  <!-- Page content begins here -->
  <h2>A Description List</h2>

  <dl>
    <dt>Coffee</dt>
    <dd>- black hot drink</dd>
    <dt>Milk</dt>
    <dd>- white cold drink</dd>
  </dl>
  <!-- Page content ends here -->
  </body>
</html>
```

Code Example: lec02b-16-HTML-description-lists.html



HTML: Table

- A table is a matching structure for displaying tabular information (row by column)
  - A web system is often a front end to a database. Most databases are relational databases with records stored as tables
- Basic components:
  - Table **cells/data** (`td`): the basic structural unit of a table, containing data
  - Table **rows** (`tr`): one horizontal row of cells
  - Table **head** (`th`): the head of each column



HTML: Table (Example)

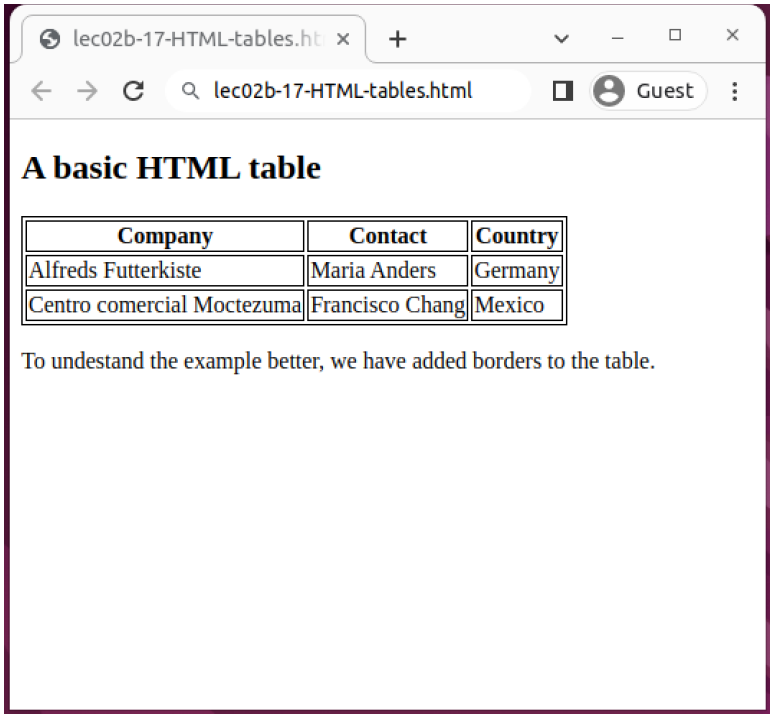
```
<!DOCTYPE html>
<html>
<style>
table, th, td {
  border:1px solid black;
}
</style>
<body>

<h2>A basic HTML table</h2>

<table>
  <tr>
    <th>Company</th>
    <th>Contact</th>
    <th>Country</th>
  </tr>
  <tr>
    <td>Alfreds Futterkiste</td>
    <td>Maria Anders</td>
    <td>Germany</td>
  </tr>
  <tr>
    <td>Centro comercial Moctezuma</td>
    <td>Francisco Chang</td>
    <td>Mexico</td>
  </tr>
</table>

<p>To undestand the example better, we have added borders to the table.</p>

</body>
</html>
```

To be covered In CSS part

Table head

Table row

Table row



Code Example: lec02b-17-HTML-tables.html

HTML: Table Details

- **Caption** (<caption> </caption>)
  - Provides a short description of table's purpose
  - **Only** permitted immediately after the <table> tag
  - **Only** one <caption> element in a table
- **Header** (<thead> </thead>)
  - Contains heading information of each column
  - Can have multiple rows inside, i.e., with more than one <tr>
  - <th> is usually used in header, instead of <td>
- **Body** (<tbody> </tbody>)
  - Contains the rows showing table's contents
- **Footer** (<tfoot> </tfoot>)
  - Contains table's footnote information
  - Useful for a summary

```html
<!DOCTYPE html>
<html>
<head>
<style>
table, th, td {
  border: 1px solid black;
}
</style>
</head>
<body>

<h1>The caption,thead, tbody, and tfoot elements</h1>

<table>
    <caption>Monthly savings</caption>
    <thead>
        <tr>
        <th>Month</th>
        <th>Savings</th>
        </tr>
    </thead>
    <tbody>
        <tr>
        <td>January</td>
        <td>$100</td>
        </tr>
        <tr>
        <td>February</td>
        <td>$80</td>
        </tr>
    </tbody>
    <tfoot>
        <tr>
        <td>Sum</td>
        <td>$180</td>
        </tr>
    </tfoot>
</table>

</body>
</html>
```

Code Example: lec02b-18-HTML-table-details.html

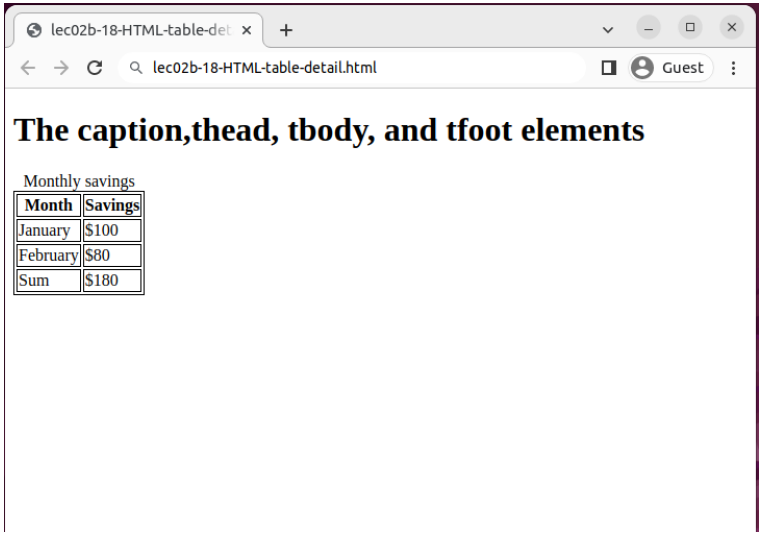


Table Details (2)

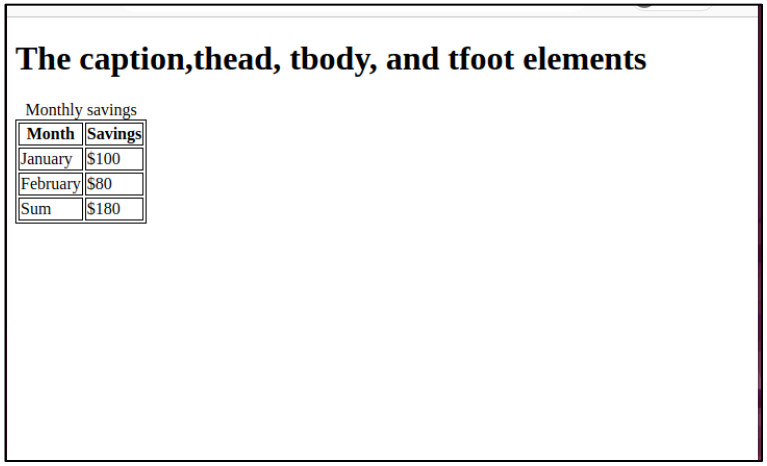
Advantages of separating table heading, body, and footer

- Standard format is set by default (e.g., <thead> sets the heading font bold)
- Easy to define and manage the styles for each section
- Storing data in a more reasonable way (e.g., data cells are within a hierarchy)

How to add another row?

How about another column?

```
<!DOCTYPE html>
<html>
<head>
<style>
table, th, td {
  border: 1px solid black;
}
</style>
</head>
<body>

<h1>The caption,thead, tbody, and tfoot elements</h1>

<table>
    <caption>Monthly savings</caption>
    <thead>
        <tr>
        <th>Month</th>
        <th>Savings</th>
        </tr>
    </thead>
    <tbody>
        <tr>
        <td>January</td>
        <td>$100</td>
        </tr>
        <tr>
        <td>February</td>
        <td>$80</td>
        </tr>
    </tbody>
    <tfoot>
        <tr>
        <td>Sum</td>
        <td>$180</td>
        </tr>
    </tfoot>
</table>

</body>
</html>
```



HTML: Table Cells Merging

```
<table style="width:80%">
<tr>
  <td colspan="2">1</td>
  <td>3</td>
</tr>
<tr>
  <td>4</td>
  <td>5</td>
  <td rowspan="2">6</td>
</tr>
<tr>
  <td>7</td>
  <td>8</td>
</tr>
</table>
<br />
<table style="width:80%">
<tr>
  <td colspan="2">1</td>
  <td>2</td>
  <td>3</td>
</tr>
<tr>
  <td>4</td>
  <td>5</td>
  <td rowspan="2">6</td>
</tr>
<tr>
  <td>7</td>
  <td>8</td>
  <td>9</td>
</tr>
</table>
```

Code Example: lec02b-19-HTML-table-cell-merging.html

- It is common to have cells merged to form a larger cell across *columns* or *rows*
- The **attributes** `rowspan` and `colspan` can be used

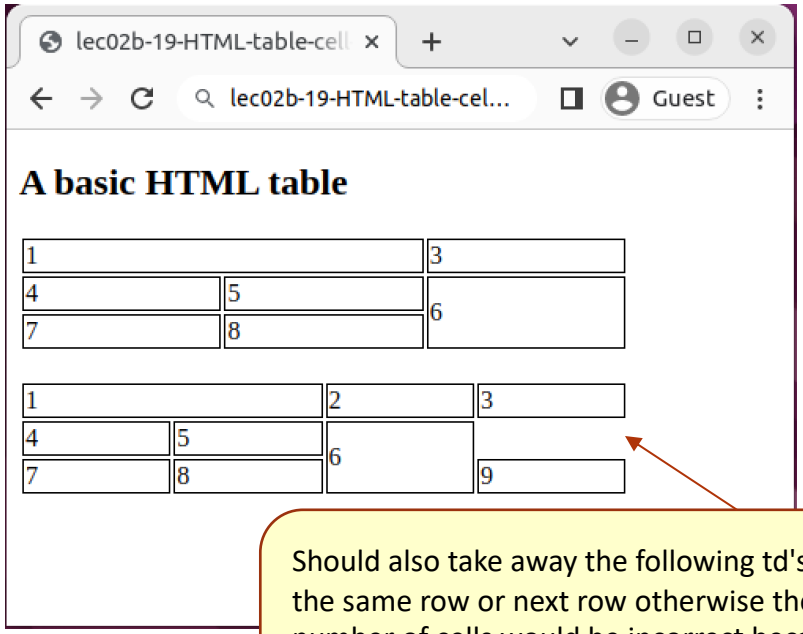
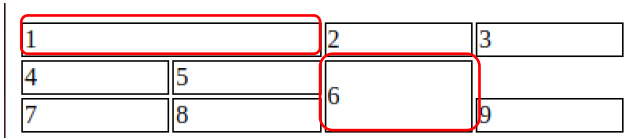


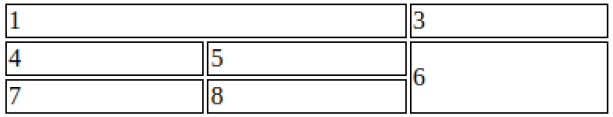
Table Cells Merging (2)

```
<table style="width:80%">
<tr>
  <td colspan="2">1</td>
  <td>2</td>
  <td>3</td>
</tr>
<tr>
  <td>4</td>
  <td>5</td>
  <td rowspan="2">6</td>
</tr>
<tr>
  <td>7</td>
  <td>8</td>
  <td>9</td>
</tr>
</table>
```



How to make the table rows/columns correct?

Method 1: 3 x 3 table




```
<table style="width:80%">
<tr>
  <td colspan="2">1</td>
  <td>3</td>
</tr>
<tr>
  <td>4</td>
  <td>5</td>
  <td rowspan="2">6</td>
</tr>
<tr>
  <td>7</td>
  <td>8</td>
</tr>
</table>
<br />
```

Table Cells Merging (3)

```
<table style="width:80%">
<tr>
  <td colspan="2">1</td>
  <td>2</td>
  <td>3</td>
</tr>
<tr>
  <td>4</td>
  <td>5</td>
  <td rowspan="2">6</td>
</tr>
<tr>
  <td>7</td>
  <td>8</td>
  <td>9</td>
</tr>
</table>
```

A basic HTML table

1		3
4	5	6
7	8	

1		2	3
4	5	6	
7	8		9

How to make the table rows/columns correct?

Method 1: 3 x 4 table

1		2	3
4	5	6	6+
7	8		9

```
<table style="width:80%">
    <tr>
        <td colspan="2">1</td>
        <td>2</td>
        <td>3</td>
    </tr>
    <tr>
        <td>4</td>
        <td>5</td>
        <td rowspan="2">6</td>
        <td>6+</td>
    </tr>
    <tr>
        <td>7</td>
        <td>8</td>
        <td>9</td>
    </tr>
</table>
```

- `rowspan/colspan = "x"` takes x cells in the table
- Make sure the number of cells is consistent across rows/columns

Critical Thinking

- How to merge cells across rows and columns at the same time?

1		2	3
4	5	6	
7	8		

HTML: Table Good Practice

- Like “list”, it is a good practice to build in more structures in table for better control
  - The tags “`thead`”, “ `tbody`” and “ `tfoot`” are not required to make your HTML to pass validation, but help to give more structures and can have a better control in styling later
- By default, there is no border shown for a table and it is difficult to check the alignment of table cells. You may create a table (some tools do this by default) with a `border` attribute

```
<table border="1">...</table>
```

- Note that border is a style related and not be set in HTML. The **border attribute** should be taken away after testing

HTML: Table Good Practice

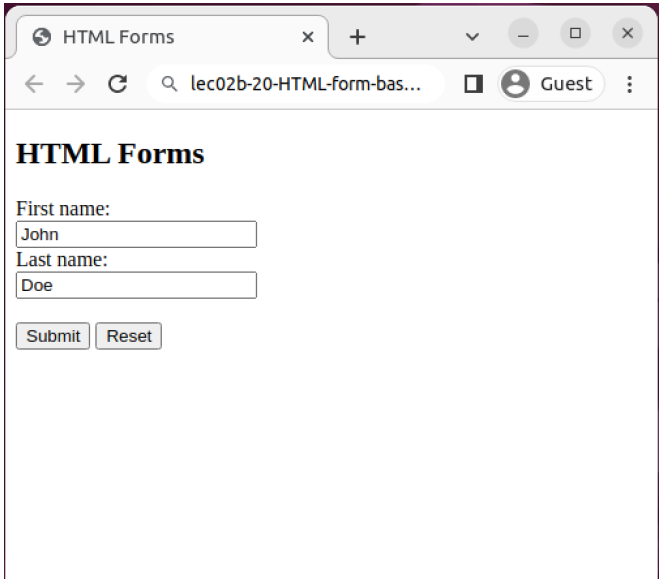
- Like list items (<li>), table cells (<td>) may contain other HTML tags too
- In the past, when CSS styling was not yet fully developed, people used table to control the layout of Web page (divide it into rows and columns - a grid)
- This approach makes the change of layout very difficult because you need to edit the number of rows and columns
- This approach should not be used anymore. Layout should be set with CSS

HTML: Form

- You may not be aware of it, but we use HTML forms all the time, e.g., any login page to a web site
- Form is important because it is the basic (standard) structure that allows user's input to be sent back to:
  - Web server
  - Other pages (because the information stored in a page will be lost when it is reloaded)

- To make a form work, two basics parts are required:
  - Structure
  - Processing script (in the server or other page)

HTML: Form Structure



attributes in the form tag:

action - the URL of the server script or target page that will receive and process the form

method - the way to communicate with the server using HTTP, Get or Post

```
<!DOCTYPE html>
<html>
<body>

<h2>HTML Forms</h2>

<form method="get" action="lec02b-23-HTML-form-script.html">
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe"><br><br>
  <input type="submit" value="Submit">
  <input type="reset" value="Reset">
</form>

</body>
</html>
```

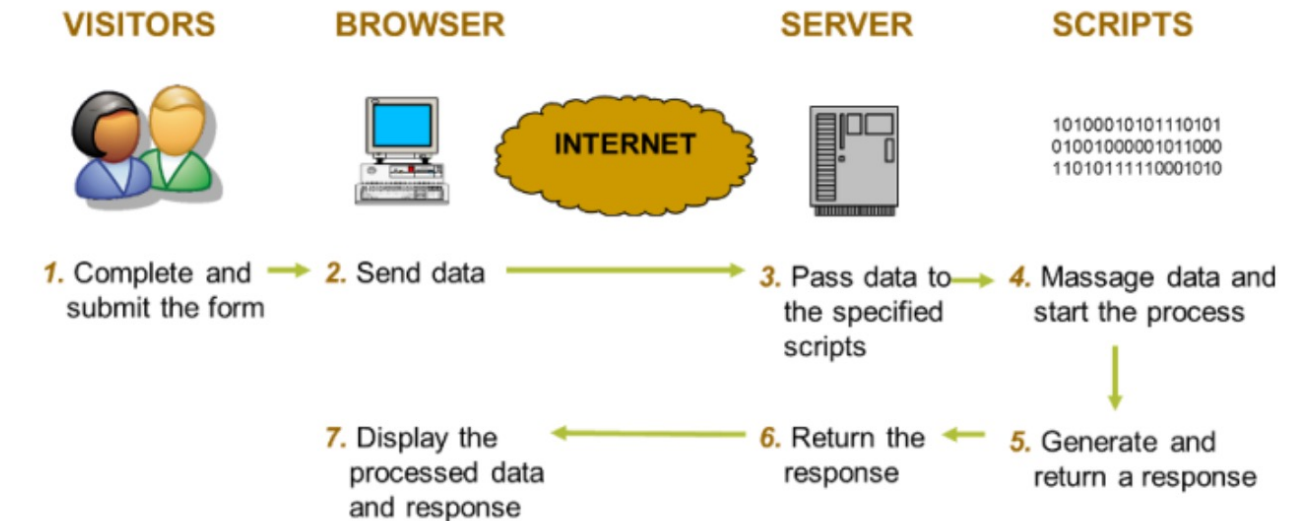
include Submit & Reset buttons, when clicked the form will be processed by browser

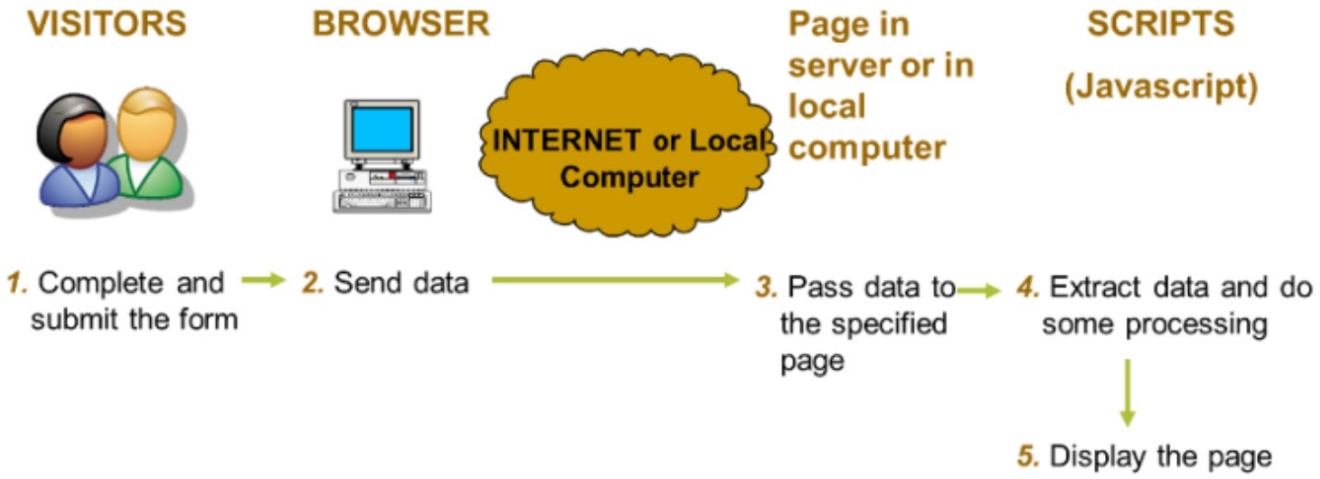
Labels and text input area, other types of inputs will be covered later

input tags inside the form tag:

Code Example: lec02b-20-HTML-form-basic-structure.html

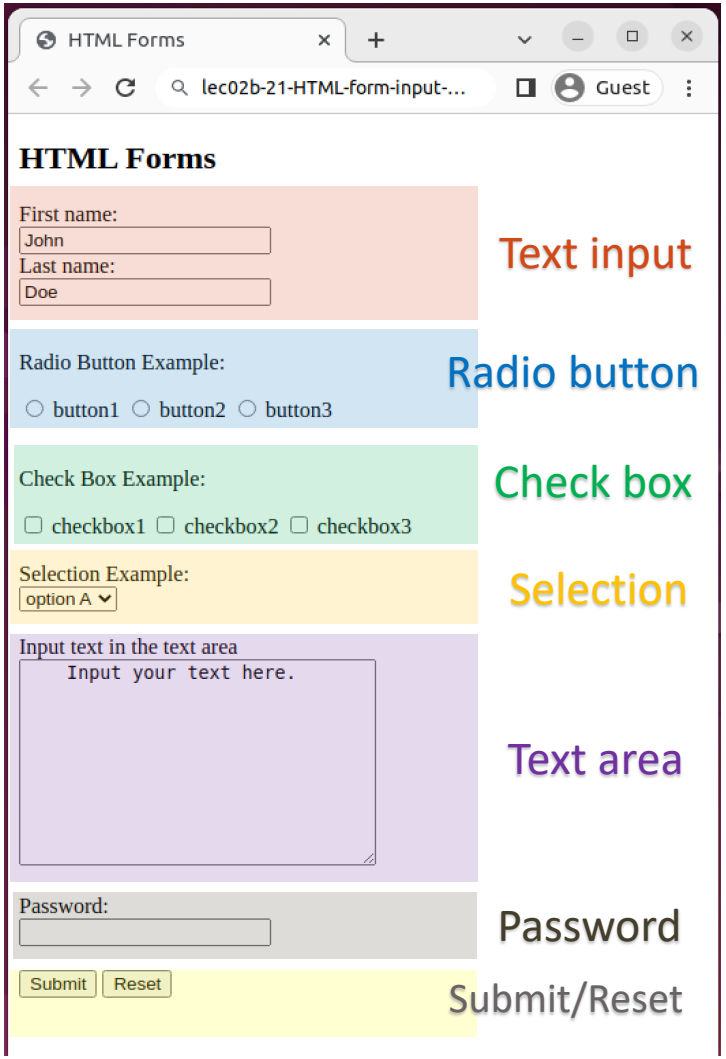
HTML: How Form Works?





HTML: Basic Form Input Element Types

```
<body>

<h2>HTML Forms</h2>

<form  method="get" action="lec02b-23-HTML-form-script.html">
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname" value="John"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname" value="Doe"><br><br>

  <p>Radio Button Example:</p>
  <input type="radio" id="button1" name="radio_button" value="button1">
  <label for="button1">button1</label>
  <input type="radio" id="button2" name="radio_button" value="button2">
  <label for="button2">button2</label>
  <input type="radio" id="button3" name="radio_button" value="button3">
  <label for="button3">button3</label><br><br>

  <p>Check Box Example:</p>
  <input type="checkbox" id="checkbox1" name="check_box" value="checkbox1">
  <label for="checkbox1"> checkbox1</label>
  <input type="checkbox" id="checkbox2" name="check_box" value="checkbox2">
  <label for="checkbox2"> checkbox2</label>
  <input type="checkbox" id="checkbox3" name="check_box" value="checkbox3">
  <label for="checkbox3"> checkbox3</label><br><br>

  <label for="selectExample">Selection Example:</label><br>
  <select id="selectExample" name="selections">
    <option value="optionA">option A</option>
    <option value="optionB">option B</option>
    <option value="optionC">option C</option>
    <option value="optionD">option D</option>
  </select><br><br>

  <label for="textinput">Input text in the text area</label><br>
  <textarea id="textinput" name="message" rows="10" cols="30">
    Input your text here.
  </textarea>

  <br><br>
  <label for="pwd">Password:</label><br>
  <input type="password" id="pwd" name="pwd"><br><br>

  <input type="submit" value="Submit">
  <input type="reset" value="Reset">
</form>

</body>
```



Code Example: lec02b-21-HTML-form-input-element-basic.html

Agenda for today's lecture

- HTML Tags
  - List
  - Table
  - Form
- Multimedia
  - Video & audio
  - Brief intro: Canvas and animation
- Doctype
- Personal webpage

HTML: Multimedia

- Great improvement in multimedia handling is the strength of HTML5
  - It is also the slow development of XHTML and its complication of processing multimedia that led to the forming of the WHATWG
  - Use of multimedia in HTML5 in fact involves CSS3 and new JavaScript API (Application Programming Interface) as well
- **Embedding** video/audio
  - This is messy in the past because of different plug-ins and media formats
  - Now simplify the element, but limit the supported formats
  - <video> & <audio>
  - Even more powerful when we learn JavaScript
- **Drawing** - Canvas: a drawing area, like , must work together with JavaScript
- **Animation**: must use JavaScript in the past, now can use style (CSS level 3)

HTML: Video&Audio

HTML5 trades extendibility for ease of use. Only 3 video formats are supported: **mp4**, **WebM** and **ogg**. The concept of plug-in is given up

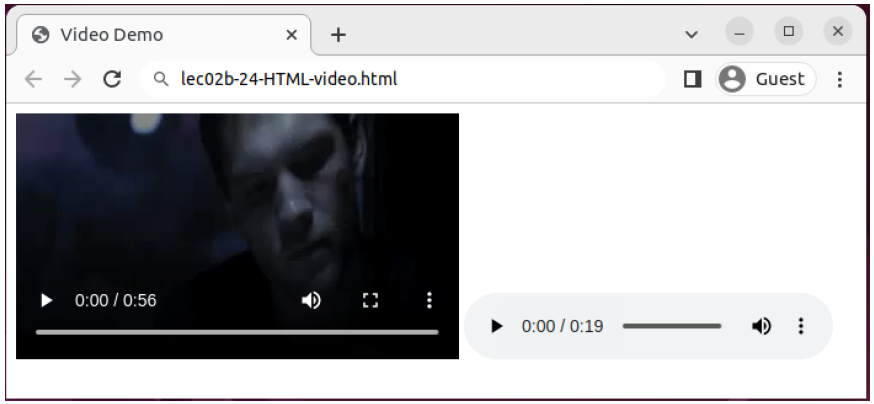
Common useful **attributes** to control the video:

- controls - show the control bar
- loop - repeat playing
- autoplay - play once fully loaded

Browser support:

Chrome - mp4, WebM & ogg
Safari - mp4
Firefox - mp4, WebM & ogg
Opera - mp4, WebM & ogg
IE - mp4

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Video Demo</title>
  </head>
  <body>
    <div id="container">
      <video src="../video/trailer.mp4" type="video/mp4"
             controls="controls"
             autoplay="autoplay">
      </video>
    <audio controls="controls">
      <source src="../video/75934537.mp3" />
      </audio>
    </div>
  </body>
</html>
```



Code Example: lec02b-24-HTML-video.html

HTML: Cross Browser Support

- **Not** all browsers support all the three formats. We need to consider providing multiple formats in order to support common browsers (although mp4 is supported by all)
- Use the source attribute can handle cross browser support

```
<video controls="controls" autoplay>
    <source src="../video/wonders.ogg" type="video/ogg">
    <source src="../video/wonders.mp4" type="video/mp4">
      <h2>Your browser does not support this video format</h2>
</video>
```

- Fall back - the browser will process the multiple sources one by one until it recognize a supported format
  - If no source format is supported, the remaining coding/HTML will be shown. This is called fall back code

Additional Markup

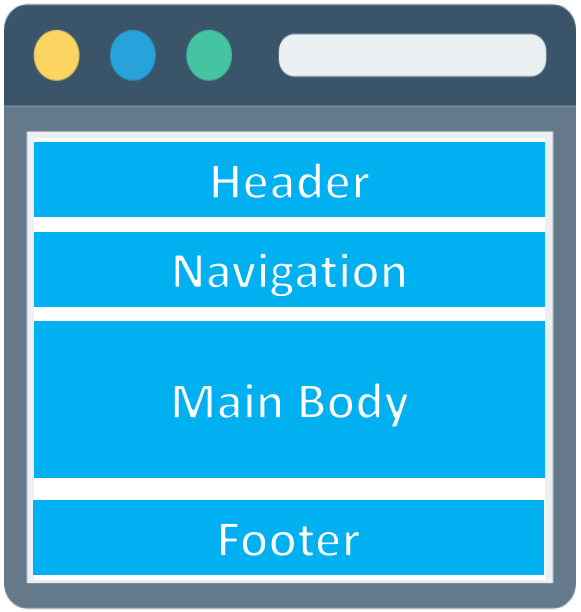
- `<iframe>` defines a window to show other information, which is short for *inline frame*
  - width and height attributes
  - src attribute: URL of the page to be shown in the frame
    - E.g., Google Maps

iframe Example


```
<iframe
    src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d3690.
    4487292240256!2d114.17022935097047!3d22.
    33667928523252!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.
    1!3m3!1m2!1s0x3404073400f3ef35%3A0xeb61704ffb0ba959!2sCity%20Unive
    rsity%20of%20Hong%20Kong!5e0!3m2!1sen!2shk!4v1663035905786!5m2!1se
    n!2shk"
    width="500"
    height="400">
</iframe>
```

Code Example: lec02b-25-HTML-iframe.html

Additional HTML5 Tags



- <header></header>
  - one or more heading elements (<h1> - <h6>)
  - logo or icon
- <nav></nav>
  - a set of navigation links
- <footer></footer>
  - footer information

```
<body>
 <header class="header"></ header >
 <nav class="nav"></nav>
 <div class="main_body"></div>
 <footer class="footer"></footer>
</body>
```

what we did before:

```
<body>
 <div class="header"></div>
 <div class="nav"></div>
 <div class="main_body"></div>
 <div class="footer"></div>
</body>
```

Accessibility - Success Criteria for Multimedia

- Videos with subtitles or sign language narration
- Additional subtitles to show ambient sounds for a clearer picture for deaf people
- Media alternatives
- Extended audio description
- Audio control
- Avoid background audio

HTML: Canvas

- **Canvas** means a drawing surface

  <canvas id="surface" width="300" height"300"> </canvas>

- The tag only defines an area for drawing, actual action of drawing needs to use JavaScript (to be learned)
- Complex drawing can be done, commonly used for graphing

HTML: Animation

- Animation cannot be done by HTML, used to be done with JavaScript. It is common to refer HTML5 techniques as including those of CSS3, which provide many features for animation without JavaScript. We will learn more in CSS
- The demo gives an interesting example - moving list items <li> with pure CSS style rules

Agenda for today's lecture

- HTML Tags
  - List
  - Table
  - Form
- Multimedia
  - Video & audio
  - Brief intro: Canvas and animation
- Doctype
- Personal webpage

What Is Doctype?

- Looking back at the history of HTML, there are 4 important milestones: HTML 4.01, XML, XHTML and HTML5
- How to specify the version if we really want to?
- The web page structure consists of a declaration part and <html>. Declaration is used to differentiate the HTML version used in the page

Doctype

- ***HTML 4.01:*** No doctype line - HTML 4.01 is assumed, e.g., for old Web pages
- ***XML:*** One more line before doctype - XML
  <?xml version="1.0" encoding="UTF-8"?>
  - It gives the ability to flexibly define the page structure/grammar by separating the document type definition, rather than built-in individual browsers
- ***XHTML:*** XHTML is said to be rewriting HTML using the XML method and the set of HTML tags. The !doctype line is more complicated.

  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
  “http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

  <html xmlns="http://www.w3.org/1999/xhtml” xml:lang=“en”>

  - This actually specifies an XML document. XHTML is defined using XML, so it is an XML document. There are also different sub-versions, such as xhtml10, xhtml11, etc. The exact grammar of a (sub)version is defined in the DTD

Doctype

- **HTML5:** HTML5 simplifies the doctype line

```
<!DOCTYPE html>
<html lang="en">
```

- Sometimes, it is known as the html syntax, but it could be written as xhtml syntax too (perhaps W3C recognized HTML5 under its xhtml framework)

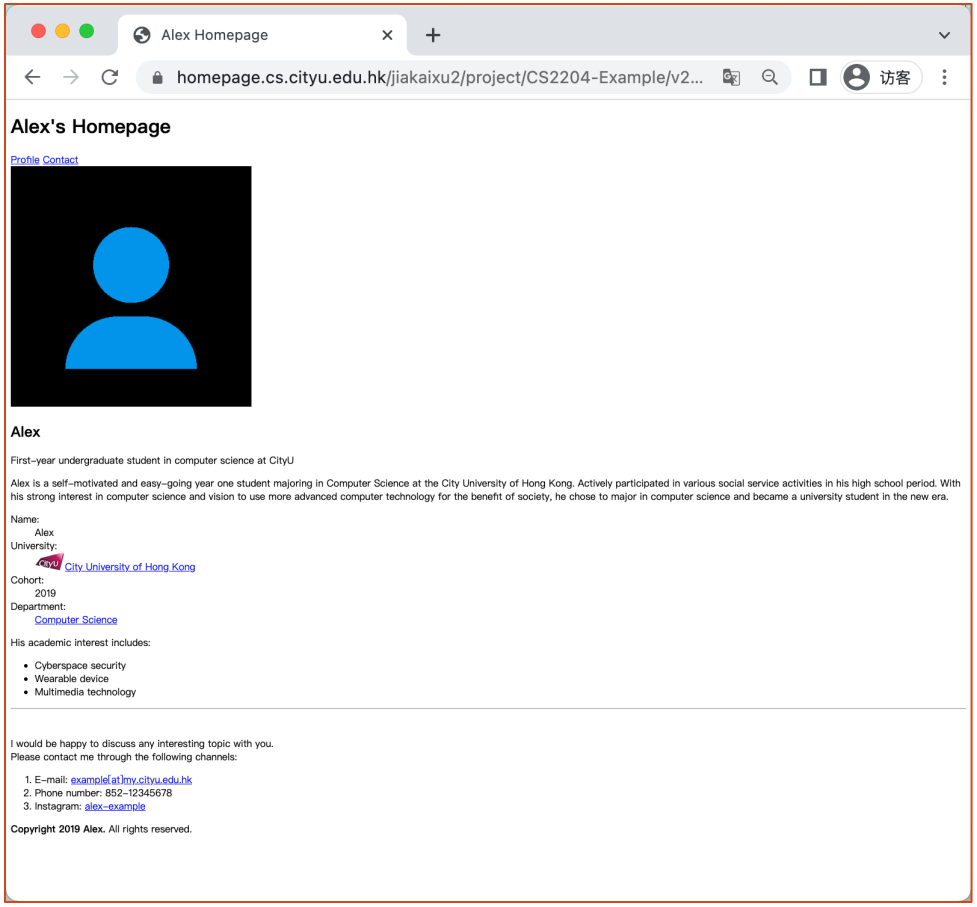
```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
```

- **Not** commonly used, may cause some subtle difference in browser rendering

Web Page Validation

- How do we know our Web page is written correctly?
  - Browser knows, but modern browsers tend to tolerate and recover errors (if possible). Since all mark-up languages have a well-defined grammar, they can be checked
- Validation: verify mark-up pages according to the version
  - Most editing tools perform syntax checking, but some subtle errors can only be found by checking against the standard or DTD (document type definition)
- W3C provides a validation page with three options:
  - by URL
  - by file upload
  - by direct input
- Convenient to use file upload in your testing, and URL will be blocked by the CS Lab firewall
- W3C validation page: https://validator.w3.org/

Personal Webpage (Version 2)



Personal Webpage (Version 2)

- **Task 1**: A simple description using <p> tag
- **Task 2:** A definition list introducing your basic information, such as name, university, department, etc.
- **Task 3:** An unordered list to introduce your academic interests
- **Task 4:** An ordered list of your contact methods
- **Task 5:** Add necessary links of websites, email and social media accounts
- **Task 6:** Add a navigator in the header to navigate to the profile/contact section using internal links