HTML

2020/21 COMP3322 Modern Technologies on WWW

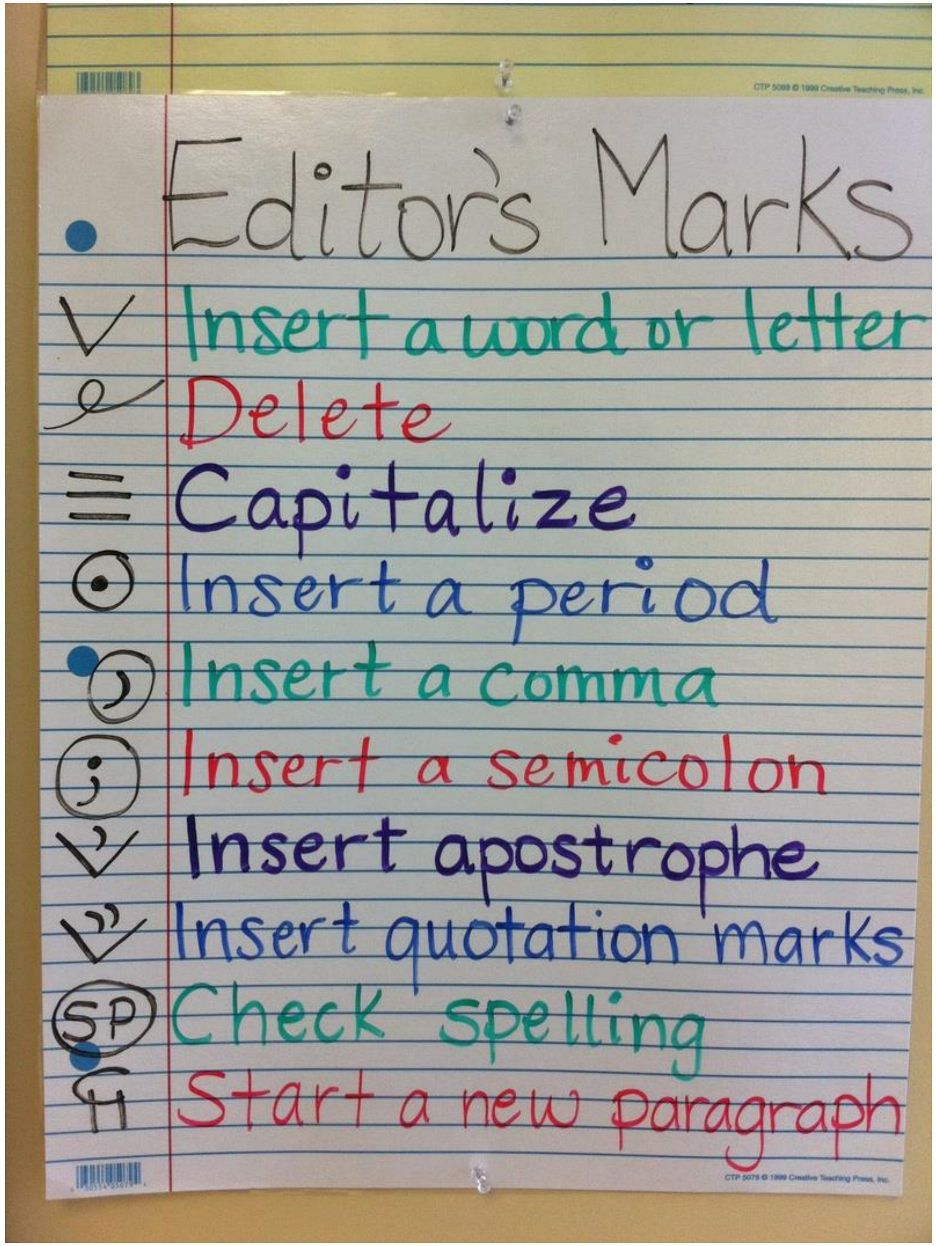
Contents

- History of HTML
- HTML Syntax and Structure of HTML
- Quick Tour of Common HTML structural Elements
- HTML Validation
- HTML5 Semantic Elements

HTML

- HTML is a type of markup languages.
- A markup language is simply a way of annotating a document in such a way
  - to make the annotations distinct from the text content being annotated
  - to indicate *information about the content*
- This "information about content" in HTML is implemented via tags (aka elements).

- Now people believe that HTML documents should **only** focus on the structure of the document.
- Information about how the content should look (i.e. presentation) is best left to CSS.



Source: Wikipedia

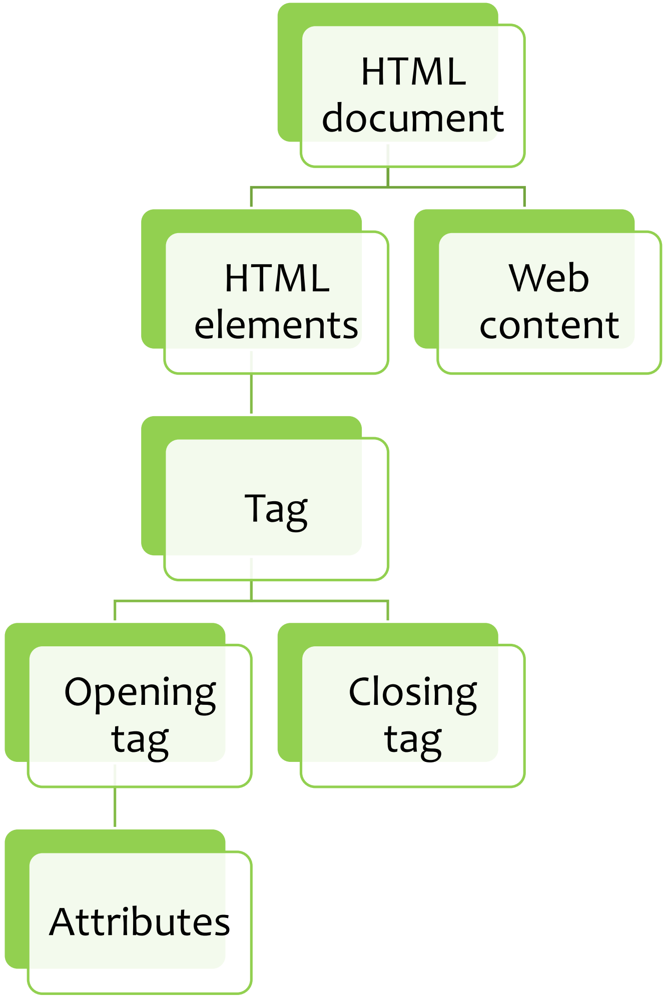
Brief History of HTML

- 1991 – the first public specification of the HTML by Tim Berners-Lee
- 1995 – HTML 2.0 was published by HTML Working Group under IETF
- Jan 1997 – HTML 3.2 was published by W3C
- Dec 1997 – HTML 4.0 was released
- Dec 1999 – HTML 4.01 was published
  - Jan 2000 – XHTML 1.0 was published, reformulation of HTML using XML
- Oct 2014 – HTML 5 was published
- Dec 2017 – HTML 5.2 was published

HTML Syntax and Structure of HTML

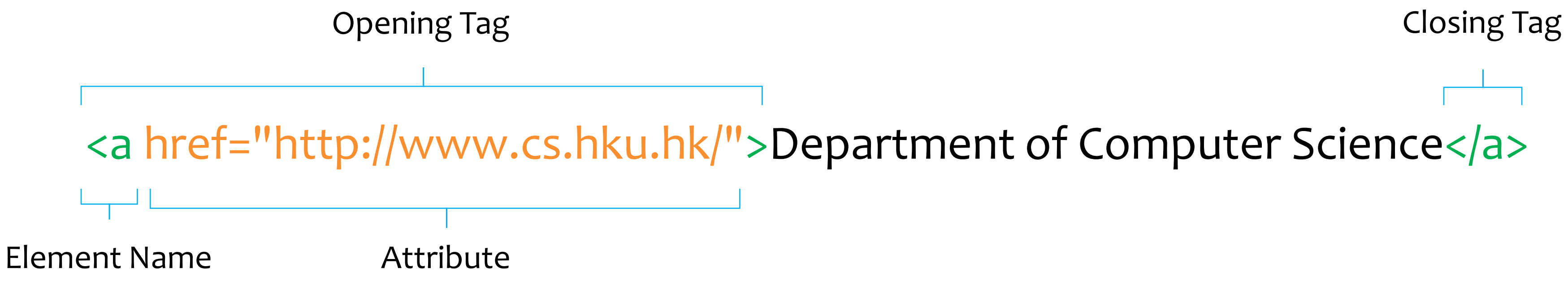
HTML Syntax: Elements and Attributes

- **HTML documents** are composed of Web content and HTML elements.
- An **HTML element** can contain text, other elements, or be empty.
- It is identified in the HTML document usually by **a pair of tags**.
  - Each HTML element **gives** the browser some **information** about the content that sits between the tags.
- HTML elements can also contain attributes. An **HTML attribute** is a ***name=value pair*** that provides more information about the HTML element.
  - *In XHTML, attribute values had to be enclosed in quotes; in HTML5, the quotes are optional.*



Elements and Attributes

- Tags act like a 'container', which give you some information (the meaning) that lies between their opening and closing tags.
- Attributes provide additional information about the contents of an element and they appear **on the opening tag** of the element.



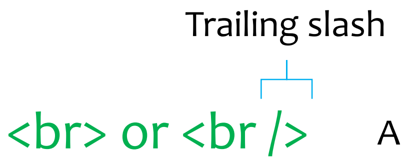
Opening Tag

Closing Tag

```
<a href="http://www.cs.hku.hk/">Department of Computer Science</a>
```

Element Name

Attribute



Trailing slash

```
<br> or <br />
```

An example of empty element - tags are not paired and with no content

HTML5 **does not** require empty elements to be closed

Elements and Attributes

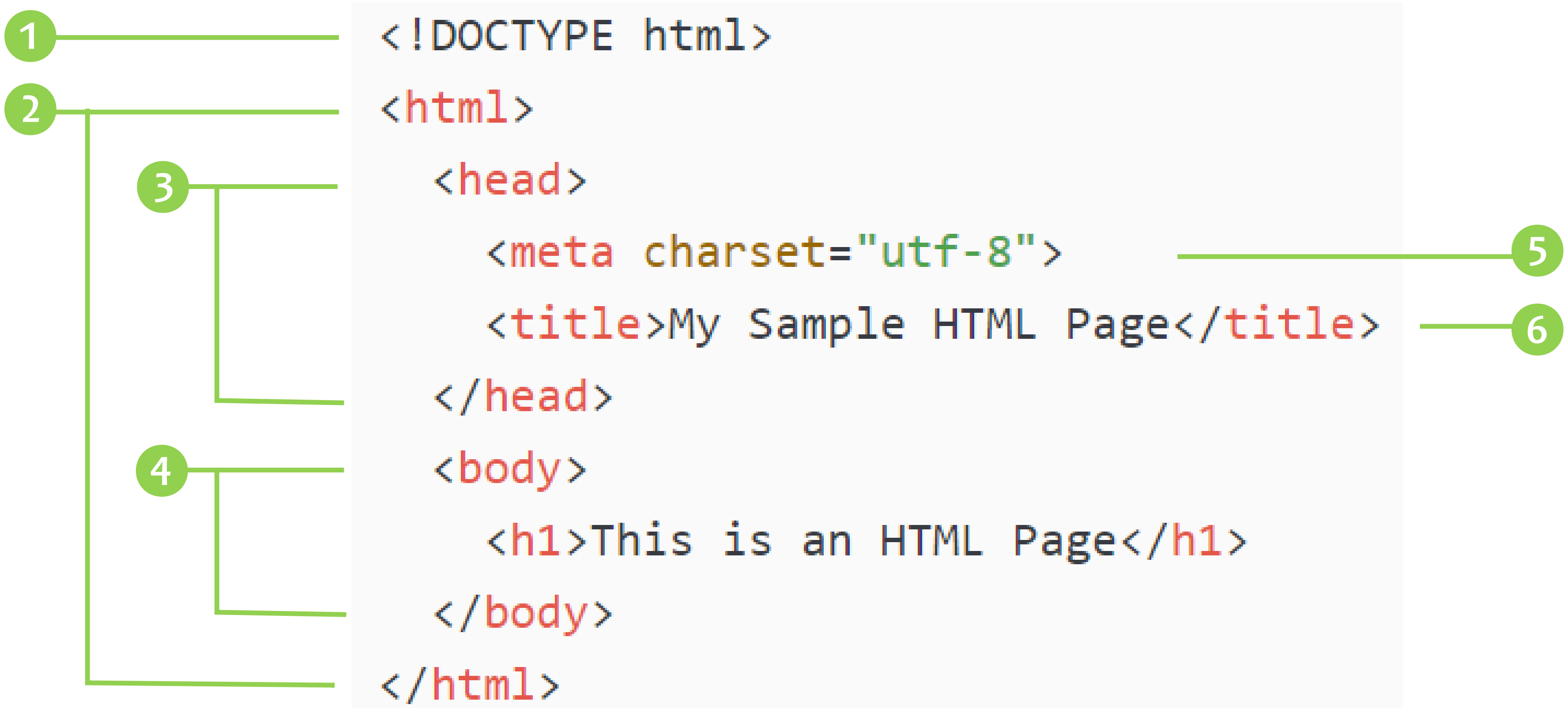
- The **attribute names** for each element **are defined** in the **HTML specifications**.
- Some attributes are required for some elements, such as the **src** and **alt** attributes in the **img** element.
- An element may have several attributes applied to it, **separated by spaces**. Their order is not important.
- Most attributes take values.
  - A value might be a number, a word, a string of text, a URL, or a measurement, depending on the purpose of the attribute.
  - Some attributes just have the name without a value.

Nesting HTML elements

- HTML element may contain other HTML elements.
  - Any elements contained within are said to be **descendents** of the parent element; likewise, any given child element, may have a variety of **ancestors**.
  - The browser is expecting a child element’s ending tag must occur before its parent’s ending tag.

```
<ul>
        <li><a href="http://www.cs.hku.hk/people/hiring/interns.jsp">Download
        Employer's Feedback Form</a></li>
</ul>
```

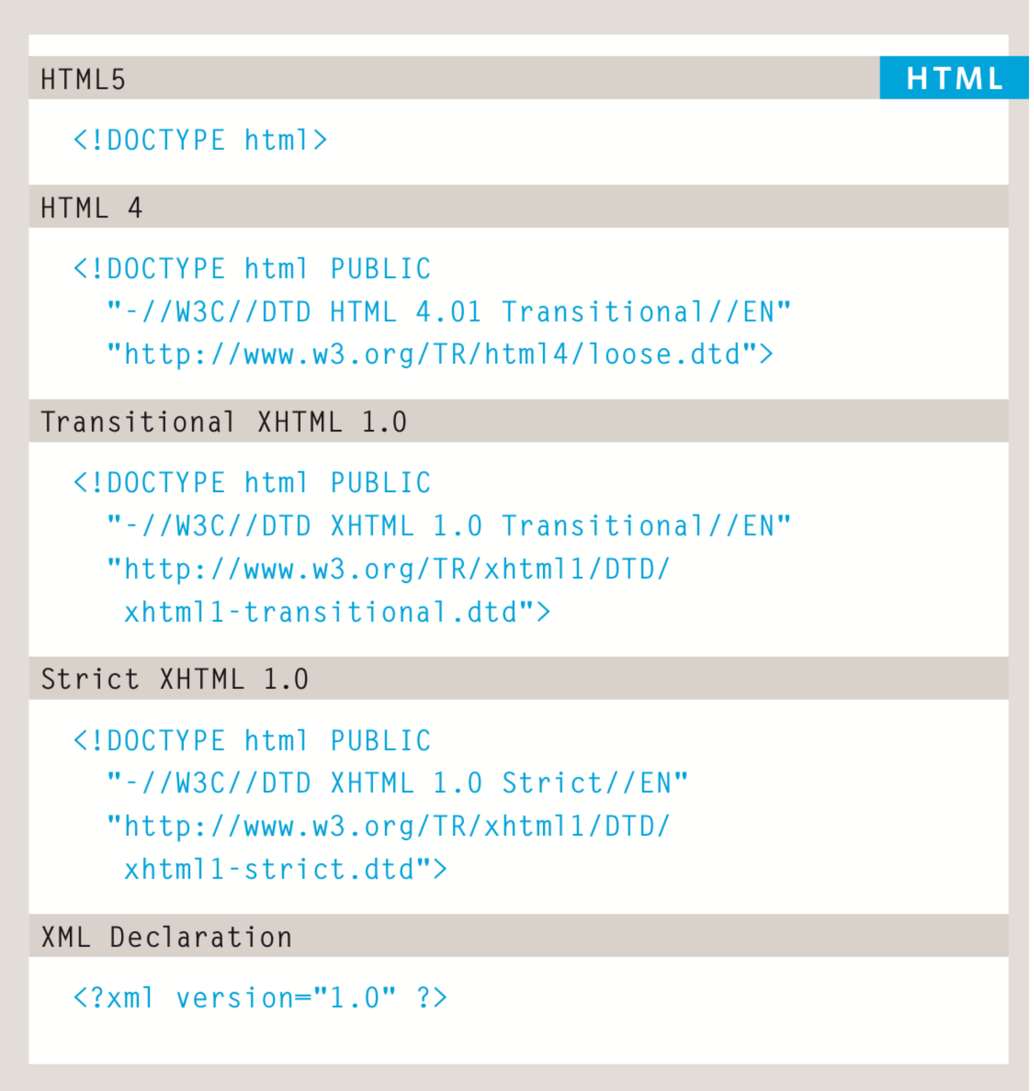
Basic Document Structure



```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My Sample HTML Page</title>
  </head>
  <body>
    <h1>This is an HTML Page</h1>
  </body>
</html>
```

<!DOCTYPE .. >

- DOCTYPE is not an HTML element
- This is a requirement imposed by HTML5 standard
- It defines the HTML version to which your page is based on, and in some cases, the Document Type Definition (DTD) that defines the specification.



HTML

HTML5

```
<!DOCTYPE html>
```

HTML 4

```
<!DOCTYPE html PUBLIC
  "-//W3C//DTD HTML 4.01 Transitional//EN"
  "http://www.w3.org/TR/html4/loose.dtd">
```

Transitional XHTML 1.0

```
<!DOCTYPE html PUBLIC
  "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/
   xhtml1-transitional.dtd">
```

Strict XHTML 1.0

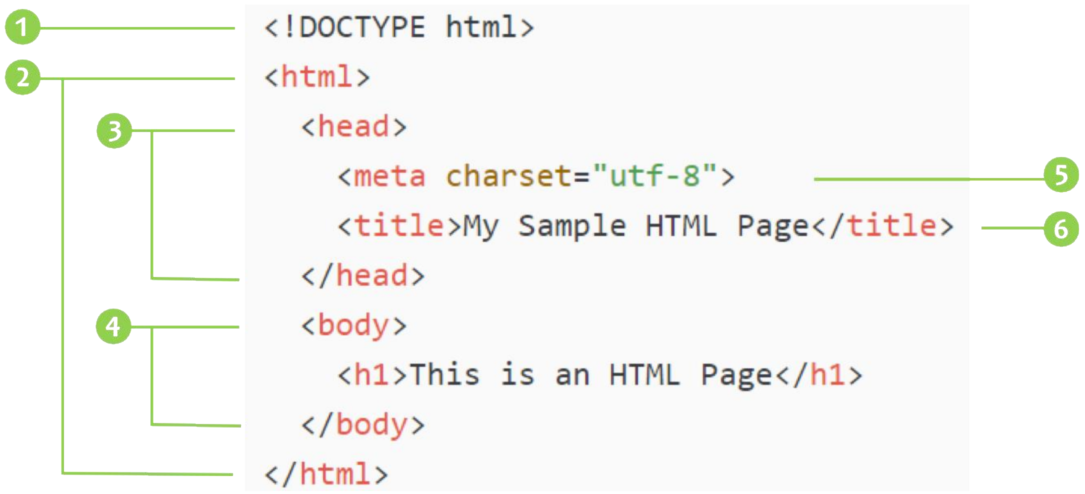
```
<!DOCTYPE html PUBLIC
  "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/
   xhtml1-strict.dtd">
```

XML Declaration

```
<?xml version="1.0" ?>
```

<html> <head> <body>

- HTML5 **does not** require the use of the <html>, <head>, and <body>.
  - In XHTML they were required, so most web authors continue to use them.
- The <html> element is sometimes called the **root element** as it contains all the other HTML elements in the document.
- The <head> contains **descriptive** elements ***about*** the document
- The <body> contains **content** that **will be displayed** by the browser.

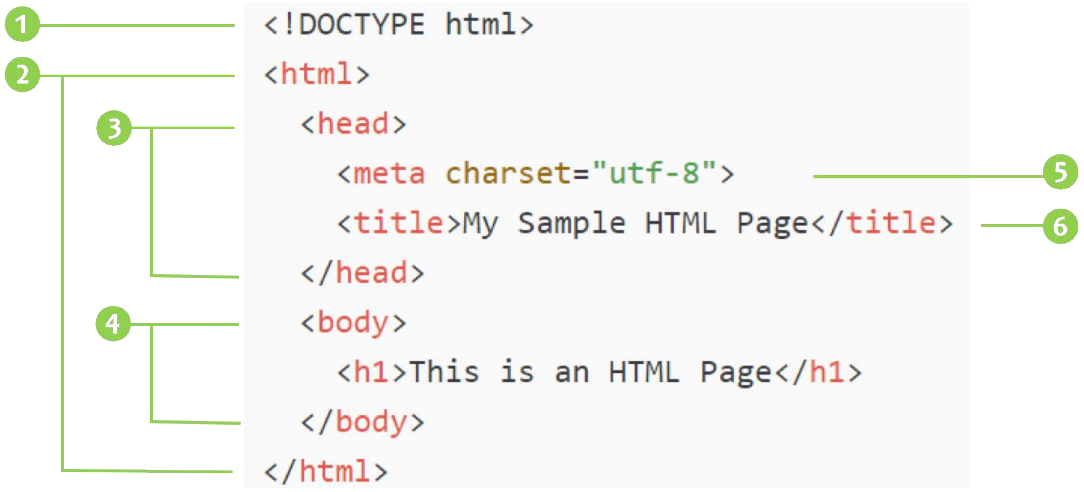


```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My Sample HTML Page</title>
  </head>
  <body>
    <h1>This is an HTML Page</h1>
  </body>
</html>
```

https://www.w3schools.com/htmL/html_head.asp

<head>

- The <head> element may contain the following elements:
  - <title> - The document’s title [**Mandatory**]
  - <link> - Specifies an **external** CSS style sheet file
  - <style> - Defines **internal style** information
  - <script> - Defines **JavaScript script** or specifies an external JavaScript file
  - <base> - Specifies base URL and base target attribute for **all relative URLs** in the document
  - <meta> - Provides all sorts of information (metadata) about the document
    - Our example declares that the character encoding for the document is UTF-8.



```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My Sample HTML Page</title>
  </head>
  <body>
    <h1>This is an HTML Page</h1>
  </body>
</html>
```

Quick Tour of Common HTML structural Elements

HTML Comment

- Add the comment between <!-- and -->
- It is a good idea to add comments to your code because, no matter how familiar you are with the page content at the time of writing it, when you come back to it later, comments will make it much easier for you to understand your own work.

Block Level Elements and Inline Elements

Block level

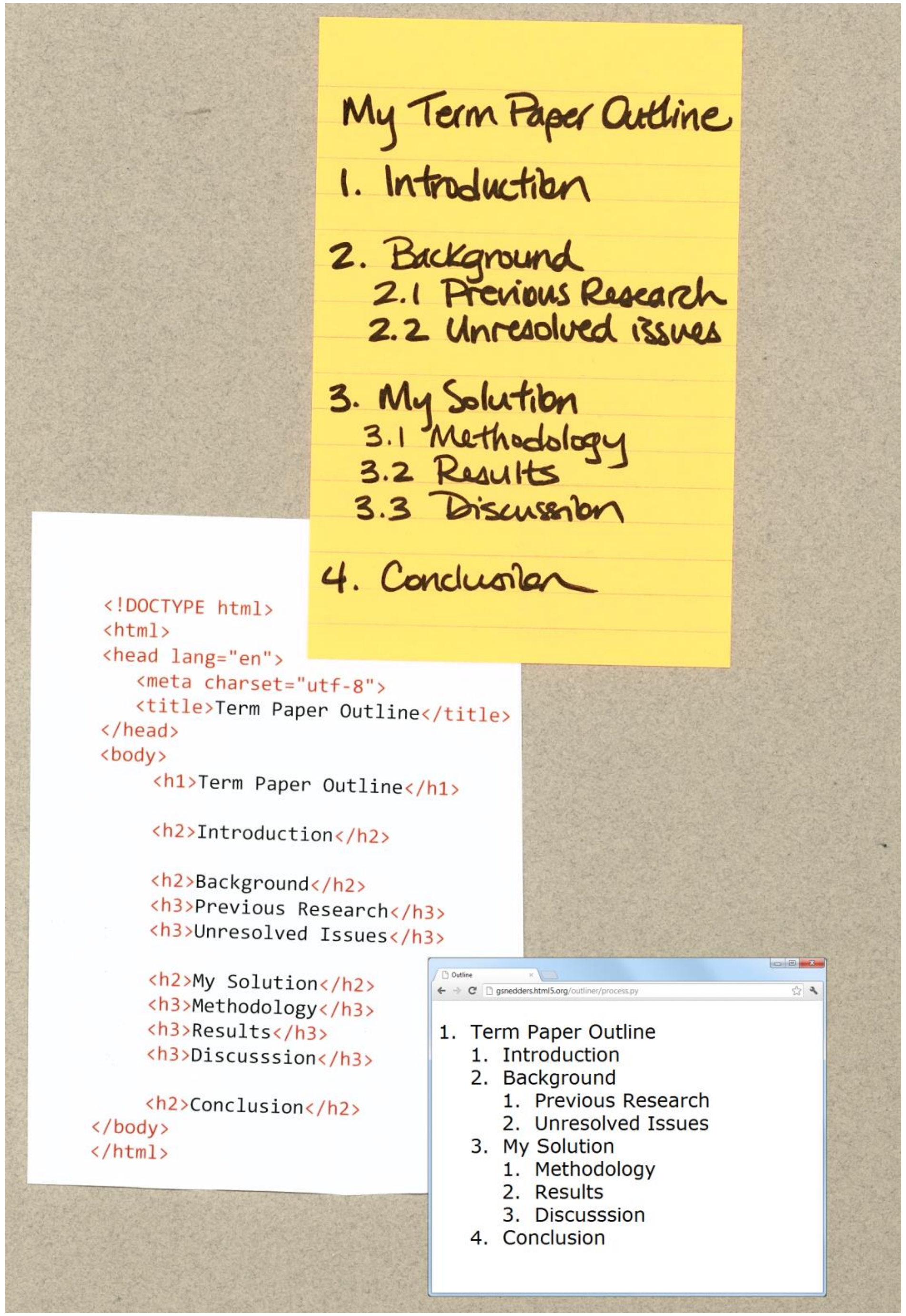
- The content contained inside the element will **start on a new line** in the browser window.
- e.g., <h1>, <p>, <div>, <ul>, <li>, <blockquote>, …
- **Acts as** a container for **styling** purpose.
  - Block elements are, by default, ***as wide as the parent container*** you place them within, and **you can modify their height and width** using CSS.

Inline

- Do not disrupt the flow of text (i.e., always **continue on the same line**).
- e.g., <img>, <a>, <b>, <i>, <em>, <strong>, …
- Also **acts as** a container for **styling** purpose.
  - Inline elements are ***only as wide as they need to display their contents***.

Headings <h1>,<h2>,..

- HTML provides **six levels** of heading (**h1, h2, h3**, …), with the higher heading number indicating a heading of less importance.
- Headings are an essential way for document authors use to show their readers the **structure** of the document.



My Term Paper Outline

1. Introduction

2. Background
   2.1 Previous Research
   2.2 Unresolved issues

3. My Solution
   3.1 Methodology
   3.2 Results
   3.3 Discussion

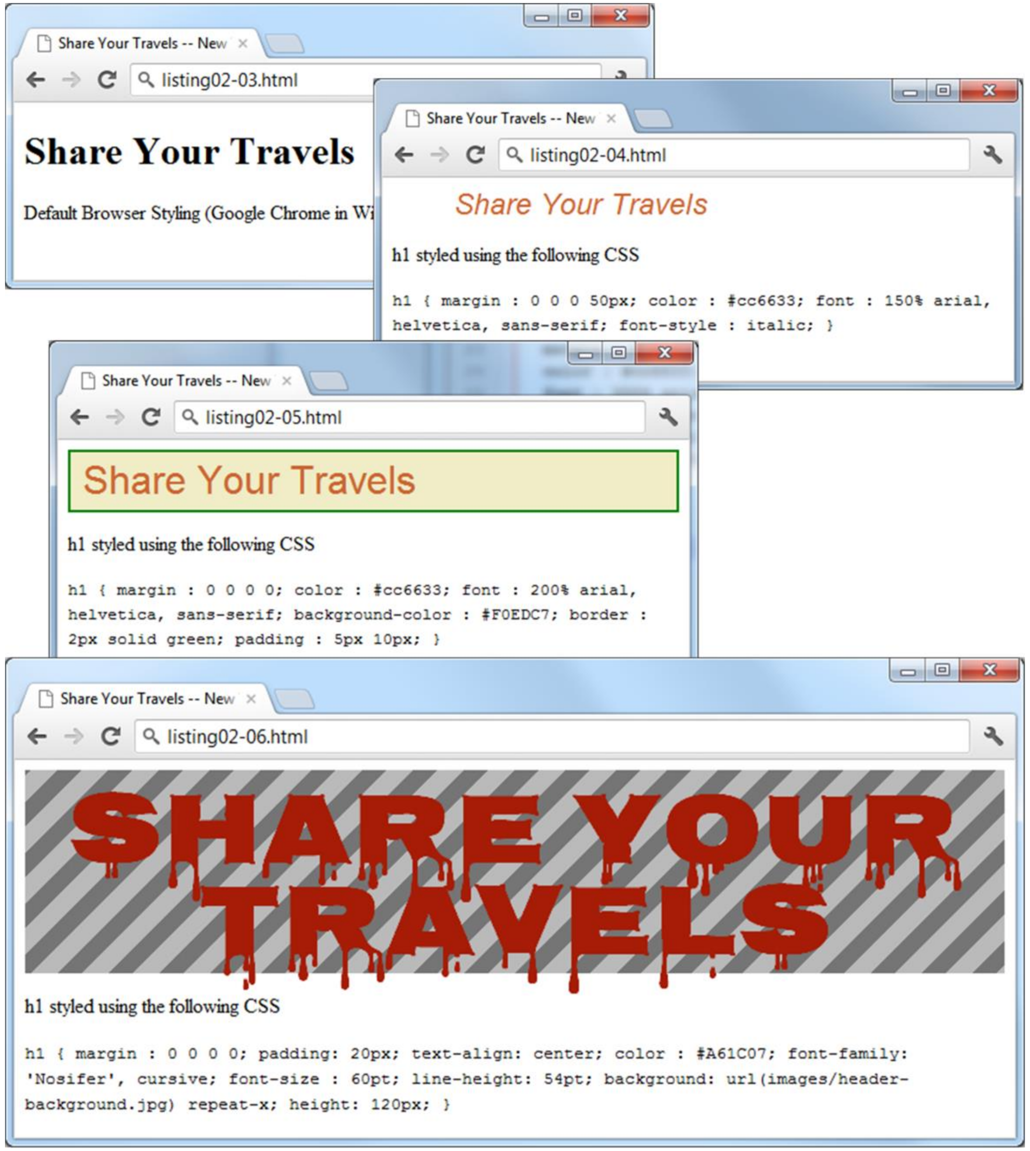
4. Conclusion

```
<!DOCTYPE html>
<html>
<head lang="en">
   <meta charset="utf-8">
   <title>Term Paper Outline</title>
</head>
<body>
    <h1>Term Paper Outline</h1>

    <h2>Introduction</h2>

    <h2>Background</h2>
    <h3>Previous Research</h3>
    <h3>Unresolved Issues</h3>

    <h2>My Solution</h2>
    <h3>Methodology</h3>
    <h3>Results</h3>
    <h3>Discusssion</h3>

    <h2>Conclusion</h2>
</body>
</html>
```

1. Term Paper Outline
   1. Introduction
   2. Background
      1. Previous Research
      2. Unresolved Issues
   3. My Solution
      1. Methodology
      2. Results
      3. Discusssion
   4. Conclusion

Headings

- The browser has its own default styling for each heading level.
- However, these are easily modified and customized via CSS.
- Headings are **semantic** elements.


Share Your Travels -- New
listing02-03.html
Share Your Travels -- New
listing02-04.html
Share Your Travels -- New
listing02-05.html
Share Your Travels -- New
listing02-06.html

Share Your Travels

Default Browser Styling (Google Chrome in Wi

Share Your Travels

h1 styled using the following CSS

```
h1 { margin : 0 0 0 50px; color : #cc6633; font : 150% arial,
helvetica, sans-serif; font-style : italic; }
```

Share Your Travels

h1 styled using the following CSS

```
h1 { margin : 0 0 0 0; color : #cc6633; font : 200% arial,
helvetica, sans-serif; background-color : #F0EDC7; border :
2px solid green; padding : 5px 10px; }
```

SHARE YOUR TRAVELS

h1 styled using the following CSS

```
h1 { margin : 0 0 0 0; padding: 20px; text-align: center; color : #A61C07; font-family:
'Nosifer', cursive; font-size : 60pt; line-height: 54pt; background: url(images/header-
background.jpg) repeat-x; height: 120px; }
```

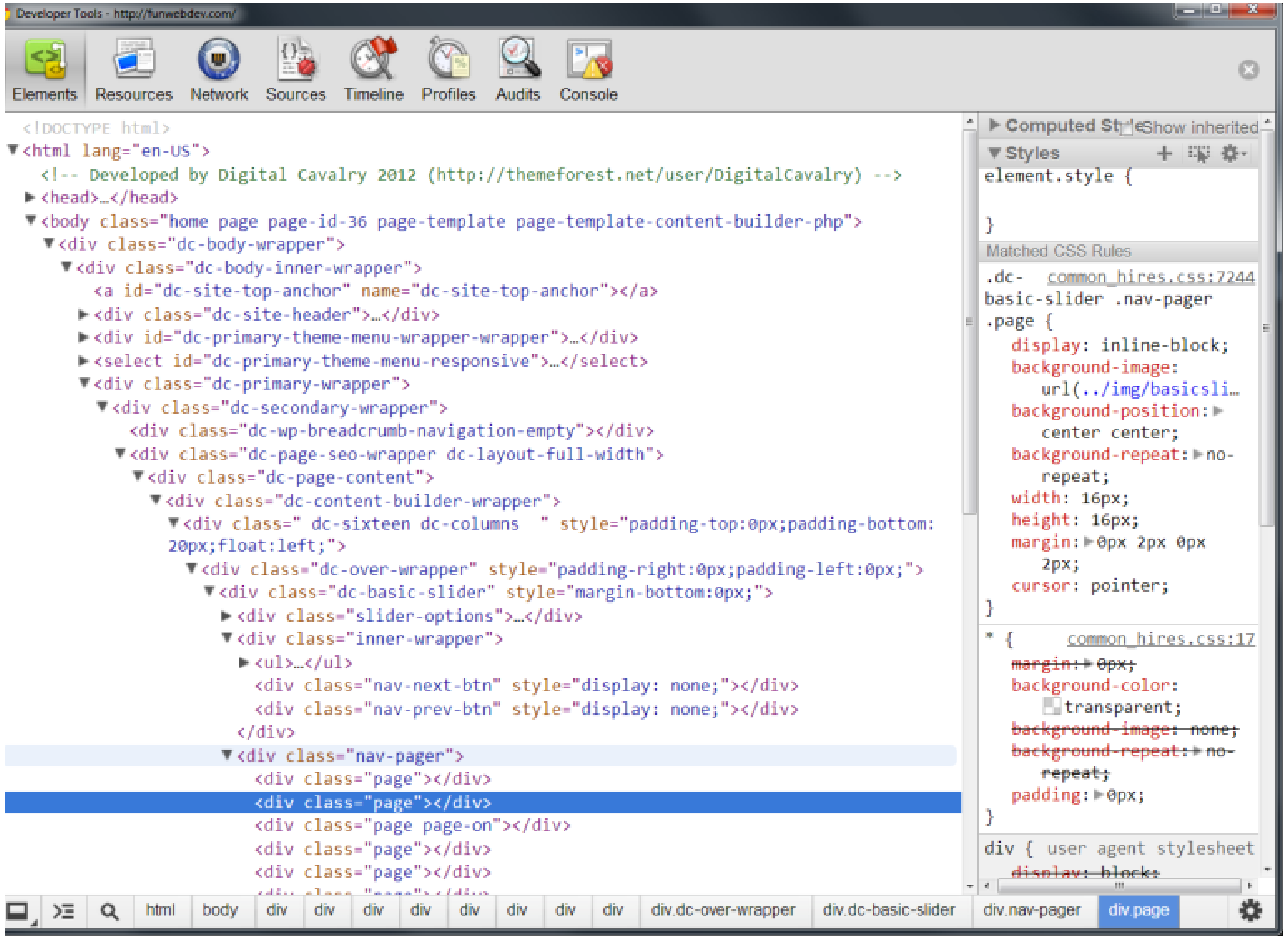
Paragraphs <p>

- Paragraphs are the most basic unit of text in an HTML document.
- The <p> tag is a container and can contain contents and other inline HTML elements

Divisions <div>

- This <div> tag is also a container element and is used to **create a logical grouping of content**
- The <div> element has **no semantic** meaning.
- It is frequently used in contemporary CSS-based layouts to mark out sections.

Using div elements



```
<!DOCTYPE html>
▼<html lang="en-US">
   <!-- Developed by Digital Cavalry 2012 (http://themeforest.net/user/DigitalCavalry) -->
  ▶<head>…</head>
  ▼<body class="home page page-id-36 page-template page-template-content-builder-php">
    ▼<div class="dc-body-wrapper">
      ▼<div class="dc-body-inner-wrapper">
         <a id="dc-site-top-anchor" name="dc-site-top-anchor"></a>
        ▶<div class="dc-site-header">…</div>
        ▶<div id="dc-primary-theme-menu-wrapper-wrapper">…</div>
        ▶<select id="dc-primary-theme-menu-responsive">…</select>
        ▼<div class="dc-primary-wrapper">
          ▼<div class="dc-secondary-wrapper">
             <div class="dc-wp-breadcrumb-navigation-empty"></div>
            ▼<div class="dc-page-seo-wrapper dc-layout-full-width">
              ▼<div class="dc-page-content">
                ▼<div class="dc-content-builder-wrapper">
                  ▼<div class=" dc-sixteen dc-columns  " style="padding-top:0px;padding-bottom:
                   20px;float:left;">
                    ▼<div class="dc-over-wrapper" style="padding-right:0px;padding-left:0px;">
                      ▼<div class="dc-basic-slider" style="margin-bottom:0px;">
                        ▶<div class="slider-options">…</div>
                        ▼<div class="inner-wrapper">
                          ▶<ul>…</ul>
                           <div class="nav-next-btn" style="display: none;"></div>
                           <div class="nav-prev-btn" style="display: none;"></div>
                         </div>
                        ▼<div class="nav-pager">
                           <div class="page"></div>
                           <div class="page"></div>
                           <div class="page page-on"></div>
                           <div class="page"></div>
                           <div class="page"></div>
                           <div class="page"></div>
```

```
element.style {

}
.dc-   common_hires.css:7244
basic-slider .nav-pager
.page {
   display: inline-block;
   background-image:
      url(../img/basicsli…
   background-position:▶
      center center;
   background-repeat:▶no-
      repeat;
   width: 16px;
   height: 16px;
   margin:▶0px 2px 0px
      2px;
   cursor: pointer;
}
* {       common_hires.css:17
   ~~margin:▶0px;~~
   background-color:
      transparent;
   ~~background-image: none;~~
   ~~background-repeat:▶no-~~
      ~~repeat;~~
   padding:▶0px;
}
div { user agent stylesheet
   ~~display: block;~~
```

- **HTML5 has a variety of new semantic elements** (which we shall examine later) that can be used to reduce somewhat the confusing mass of div within divs within divs that is so typical of contemporary web design.

Span <span>

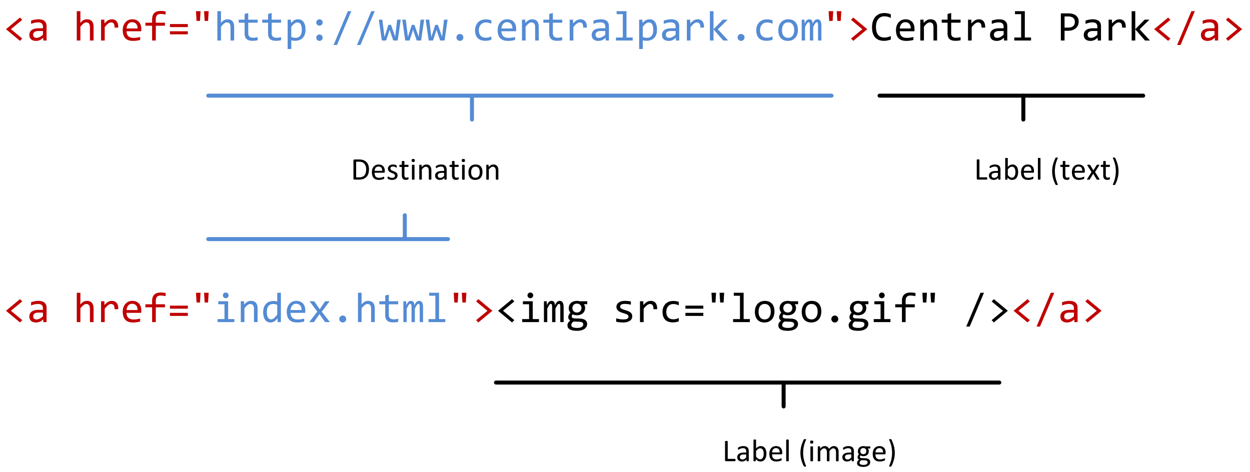
- Similar to <div>, the <span> element is often used as a container for some text. But it is an **inline element**.
- It does **not** contribute to any **meaning** at all.
- It can be used to group elements **for styling** purposes (using the **class or id attributes**), or because they share attribute values, such as lang.

```
<span style="color:red">Important</span>
```

```
<span lang="fr">Ceci est un paragraphe.</span>
```

Links <a>

- Links are created using the <a> element (the “a” stands for anchor).
- A link has two main parts: the destination and the label.



```
<a href="http://www.centralpark.com">Central Park</a>
```

Destination

Label (text)

```
<a href="index.html"><img src="logo.gif" /></a>
```

Label (image)

Types of Links

Link to external site

```
<a href="http://www.centralpark.com">Central Park</a>
```

Link to resource on external site

```
<a href="http://www.centralpark.com/logo.gif">Central Park</a>
```

Link to another page on same site as this page

```
<a href="index.html">Home</a>
```

Link to another place on the same page

```
<a href="#top">Go to Top of Document</a>
```

Link to specific place on another page

```
<a href="productX.html#reviews">Reviews for product X</a>
```

Link to email

```
<a href="mailto://person@somewhere.com">Someone</a>
```

Link to javascript function

```
<a href="javascript://OpenAnnoyingPopup();">See This</a>
```

Link to telephone (automatically dials the number
when user clicks on it using a smartphone browser)

```
<a href="tel:+18009220579">Call toll free (800) 922-0579</a>
```

URL Relative Referencing

- When referencing a resource that is **on the same server** as your HTML document, then you can use briefer relative referencing. If the URL does not include the “http://” then the browser will request the **current server** for the file.
- Pathnames on the web follow **Unix conventions**.
  - Forward slashes (“/”) are used to separate directory names from each other and from file names.
  - Double-periods (“..”) are used to reference a directory “above” the current one in the directory tree.

Images

Specifies the URL of the image to display
(note: uses standard relative referencing)

Text in title attribute will be displayed in a popup
tool tip when user moves mouse over image.

```
<img src="images/central-park.jpg" alt="Central Park"  title="Central Park" width="80" height="40" />
```

Text in alt attribute provides a brief
description of image’s content for users who
are unable to see it.

Specifies the width and height of
image in pixels.

- For purely decorative images, such as background gradients and patterns, logos, border art, and so on, it makes semantic sense to keep such images out of the markup and in CSS where they more rightly belong.
- But **when the images are content**, such as in the images in a gallery or the image of a product in a product details page, then the <img> tag is **the semantically appropriate approach**.

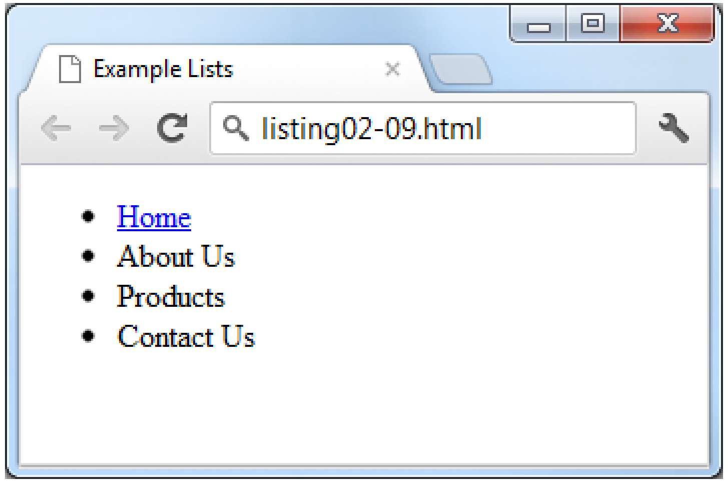
Lists

- HTML provides three types of lists
  - **Unordered lists**. Collections of items in no particular order; these are by default rendered by the browser as a **bulleted list**.
  - **Ordered lists**. Collections of items that have a set order; these are by default rendered by the browser as a **numbered list**.
  - **Description lists**. Collection of items and their associated descriptions, such as **terms and definitions**, or **questions and answers**. These tend to be used infrequently. Perhaps the most common example would be a FAQ list.

Lists

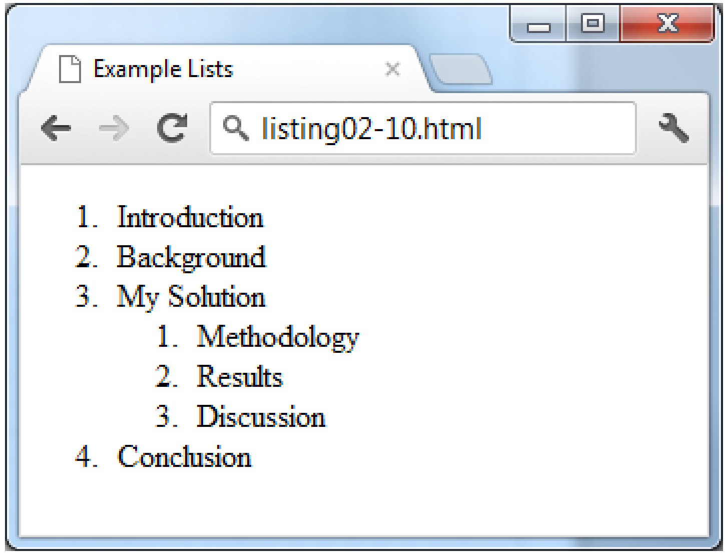
Notice that the list item element can contain other HTML elements

```
<ul>
   <li><a href="index.html">Home</a></li>
   <li>About Us</li>
   <li>Products</li>
   <li>Contact Us</li>
</ul>
```



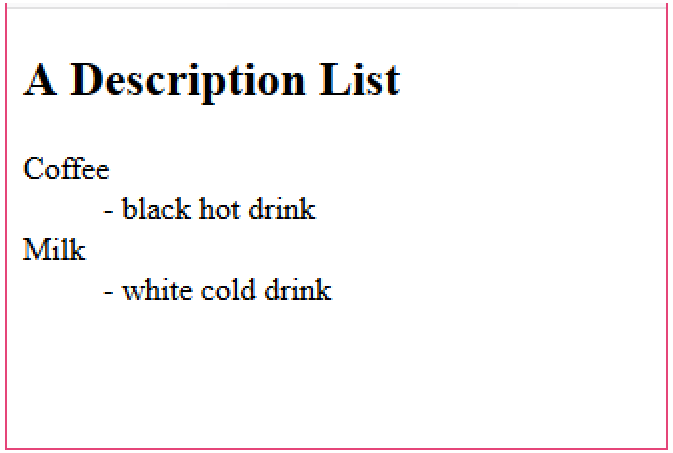
- Home
- About Us
- Products
- Contact Us

```
<ol>
   <li>Introduction</li>
   <li>Background</li>
   <li>My Solution</li>
   <li>
     <ol>
        <li>Methodology</li>
        <li>Results</li>
        <li>Discussion</li>
     </ol>
   </li>
   <li>Conclusion</li>
</ol>
```



1. Introduction
2. Background
3. My Solution
   1. Methodology
   2. Results
   3. Discussion
4. Conclusion

```
<h2>A Description List</h2>

<dl>
  <dt>Coffee</dt>
  <dd>- black hot drink</dd>
  <dt>Milk</dt>
  <dd>- white cold drink</dd>
</dl>
```



A Description List

Coffee
    - black hot drink
Milk
    - white cold drink

HTML Tables

- A table in HTML is created using the **<table>** element
- Tables can be used to display:
  - Many types of content
    - Calendars, financial data, lists, etc…
  - Any type of data
    - Images
    - Text
    - Links
    - Other tables

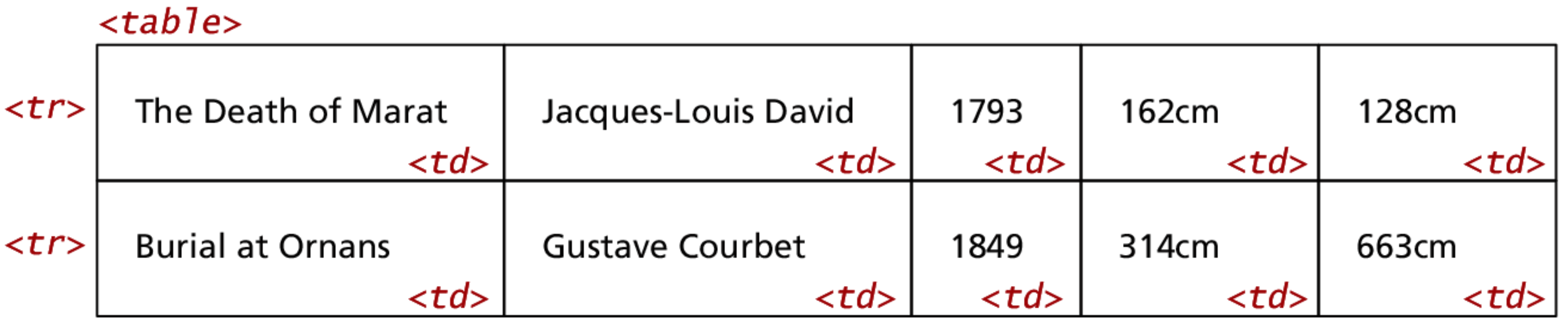
Tables Basics

- Rows and cells
  - an HTML **<table>** contains any number of rows (**<tr>**)
  - each row contains any number of table data cells (**<td>**)
  - Content goes inside of **<td></td>** tags

```
<table>
            <tr>
                        <td>The Death of Marat</td>
            </tr>
</table>
```

content

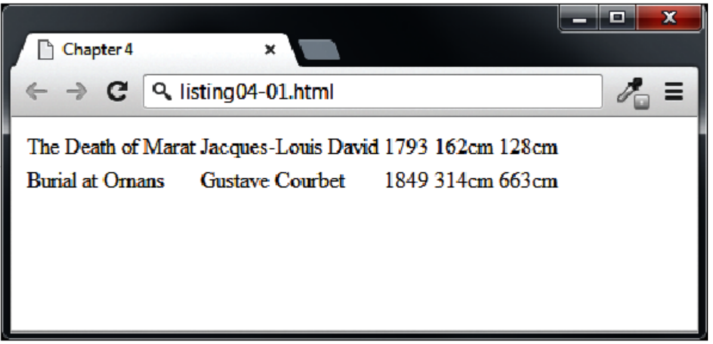
A basic Example



<table>

<tr>	The Death of Marat <td>	Jacques-Louis David <td>	1793 <td>	162cm <td>	128cm <td>
<tr>	Burial at Ornans <td>	Gustave Courbet <td>	1849 <td>	314cm <td>	663cm <td>

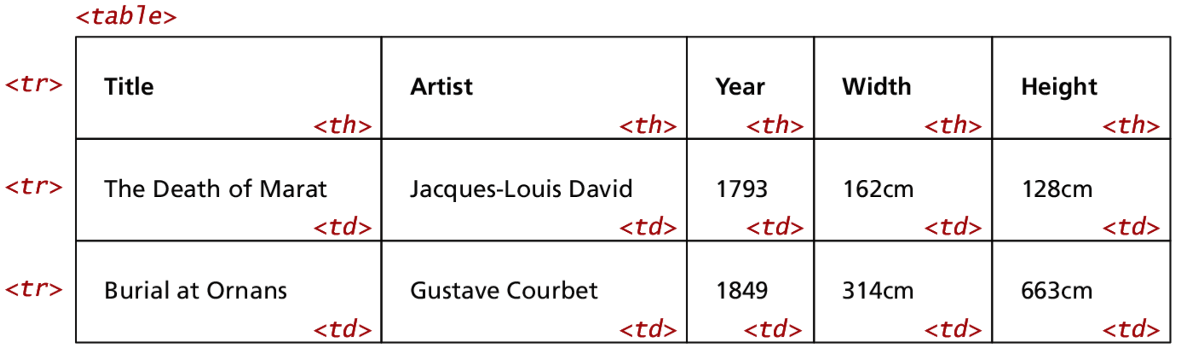
```
<table>
    <tr>
       <td>The Death of Marat</td>
       <td>Jacques-Louis David</td>
       <td>1793</td>
       <td>162cm</td>
       <td>128cm</td>
    </tr>
    <tr>
       <td>Burial at Ornans</td>
       <td>Gustave Courbet</td>
       <td>1849</td>
       <td>314cm</td>
       <td>663cm</td>
    </tr>
 </table>
```



```
The Death of Marat Jacques-Louis David 1793 162cm 128cm
Burial at Ornans      Gustave Courbet      1849 314cm 663cm
```

With Table Headings

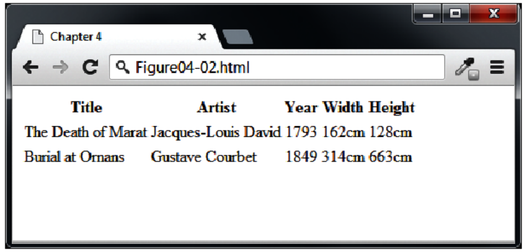
- A table heading <th> provides some **semantic info** to indicate that **a cell** is the header of a group of table cells.
  - Could be related to that row or column, which is indicated by the **scope attribute**.
  - Browsers tend to make the content within a <th> element bold
  - <th> element for accessibility (it helps those using screen readers)



<table>

<tr>	Title <th>	Artist <th>	Year <th>	Width <th>	Height <th>
<tr>	The Death of Marat <td>	Jacques-Louis David <td>	1793 <td>	162cm <td>	128cm <td>
<tr>	Burial at Ornans <td>	Gustave Courbet <td>	1849 <td>	314cm <td>	663cm <td>

```
<table>
   <tr>
      <th>Title</th>
      <th>Artist</th>
      <th>Year</th>
      <th>Width</th>
      <th>Height</th>
   </tr>
   <tr>
      <td>The Death of Marat</td>
      <td>Jacques-Louis David</td>
      <td>1793</td>
      <td>162cm</td>
      <td>128cm</td>
   </tr>
   <tr>
      <td>Burial at Ornans</td>
      <td>Gustave Courbet</td>
      <td>1849</td>
      <td>314cm</td>
      <td>663cm</td>
   </tr>
</table>
```

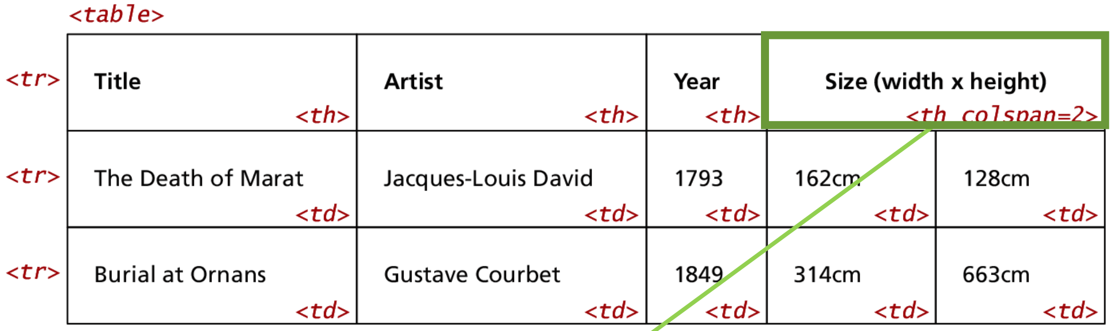


Title	Artist	Year	Width	Height
The Death of Marat	Jacques-Louis David	1793	162cm	128cm
Burial at Ornans	Gustave Courbet	1849	314cm	663cm

https://www.w3schools.com/html/html_tables.asp

Spanning Rows and Columns

- Each row must have the same number of <td> or <th> containers.



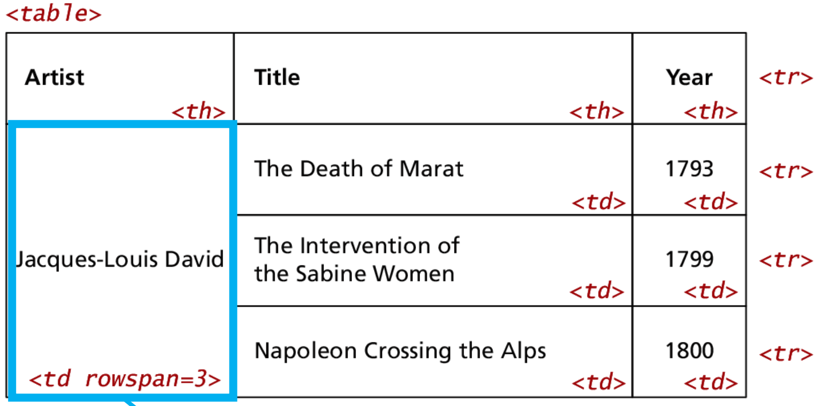
<table>

<tr>	Title <th>	Artist <th>	Year <th>	Size (width x height) <th colspan=2>	
<tr>	The Death of Marat <td>	Jacques-Louis David <td>	1793 <td>	162cm <td>	128cm <td>
<tr>	Burial at Ornans <td>	Gustave Courbet <td>	1849 <td>	314cm <td>	663cm <td>

```
<table>
  <tr>
    <th>Title</th>
    <th>Artist</th>
    <th>Year</th>
    <th colspan="2">Size (width x height)</th>
  </tr>
  <tr>
    <td>The Death of Marat</td>
    <td>Jacques-Louis David</td>
    <td>1793</td>
    <td>162cm</td>
    <td>128cm</td>
  </tr>
  ...
</table>
```

Notice that this row now only has four cell elements.

If you want a given cell to cover several columns or rows, use the **colspan or rowspan attributes.**



<table>

Artist <th>	Title <th>	Year <th>	<tr>
Jacques-Louis David <td rowspan=3>	The Death of Marat <td>	1793 <td>	<tr>
	The Intervention of the Sabine Women <td>	1799 <td>	<tr>
	Napoleon Crossing the Alps <td>	1800 <td>	<tr>

```
<table>
  <tr>
    <th>Artist</th>
    <th>Title</th>
    <th>Year</th>
  </tr>
  <tr>
    <td rowspan="3">Jacques-Louis David</td>
    <td>The Death of Marat</td>
    <td>1793</td>
  </tr>
  <tr>
    <td>The Intervention of the Sabine Women</td>
    <td>1799</td>
  </tr>
  <tr>
    <td>Napoleon Crossing the Alps</td>
    <td>1800</td>
  </tr>
  ...
</table>
```

Notice that these two rows now only have two cell elements.

Additional table tags

- <caption>
- <col>,<colgroup>
- <thead>
- <tfoot>
- <tbody>

```
<table>
```

A title for the table is good for accessibility.

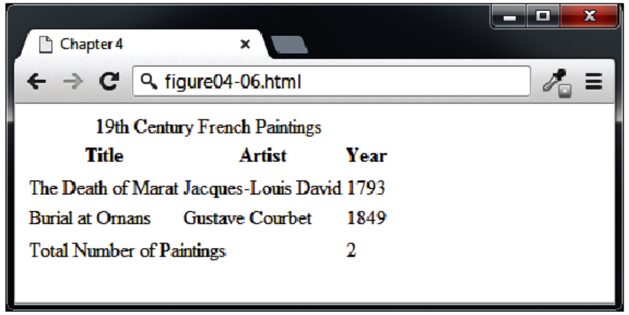
```
    <caption>19th Century French Paintings</caption>
```

These describe our columns, and can be used to aid in styling.

```
    <col class="artistName" />
    <colgroup id="paintingColumns">
        <col />
        <col />
    </colgroup>
```

Table header could potentially also include other <tr> elements.

```
    <thead>
        <tr>
            <th>Title</th>
            <th>Artist</th>
            <th>Year</th>
        </tr>
    </thead>
```



19th Century French Paintings

Title	Artist	Year
The Death of Marat	Jacques-Louis David	1793
Burial at Ornans	Gustave Courbet	1849
Total Number of Paintings		2

Yes, the table footer comes *before* the body.

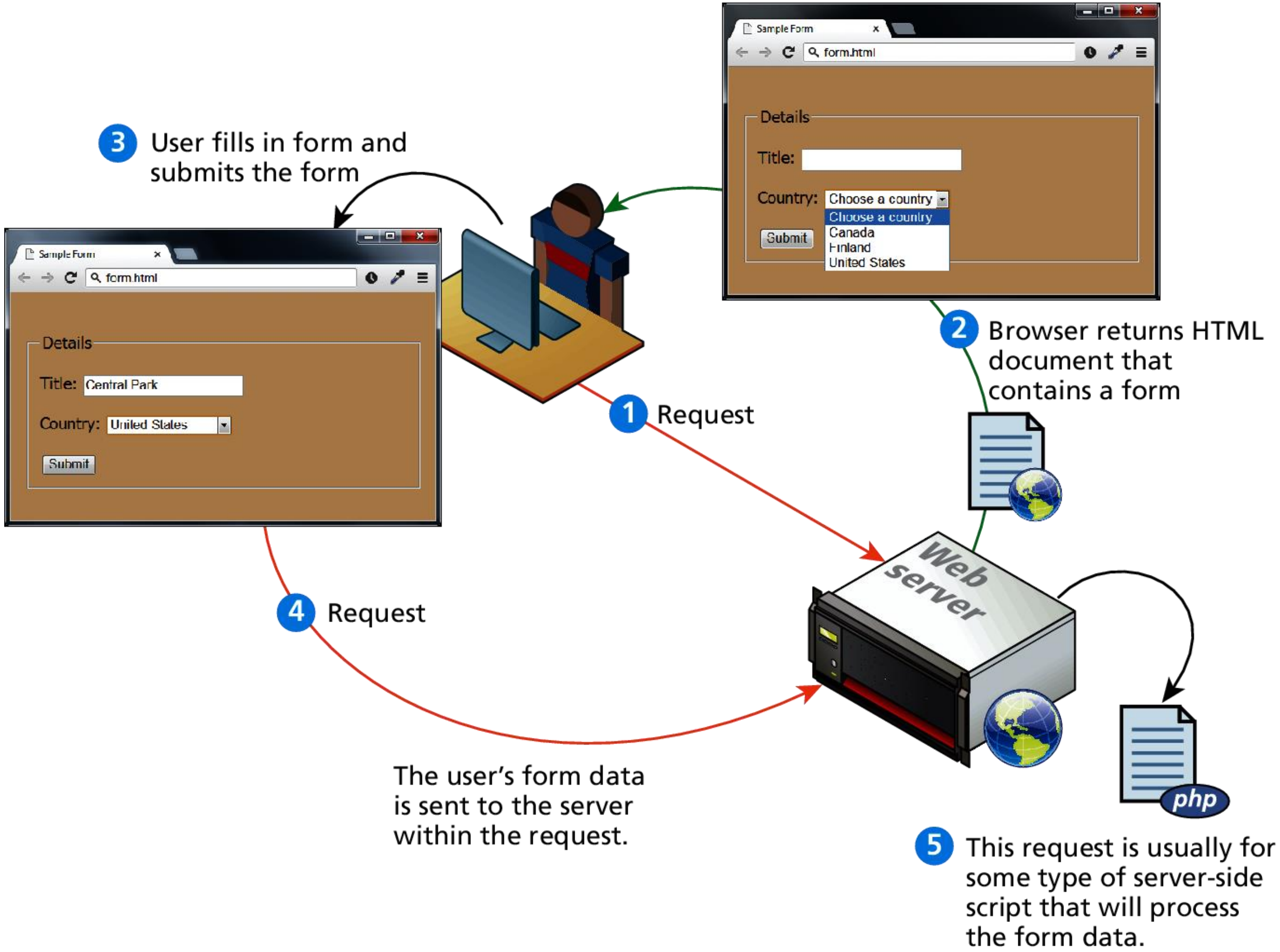
```
    <tfoot>
        <tr>
            <td colspan="2">Total Number of Paintings</td>
            <td>2</td>
        </tr>
    </tfoot>
```

Potentially, with styling the browser can scroll this information, while keeping the header and footer fixed in place.

```
    <tbody>
        <tr>
            <td>The Death of Marat</td>
            <td>Jacques-Louis David</td>
            <td>1793</td>
        </tr>
        <tr>
            <td>Burial at Ornans</td>
            <td>Gustave Courbet</td>
            <td>1849</td>
        </tr>
    </tbody>

</table>
```

HTML Forms

- Forms provide us a way to collect information from visitors (by uploading) to our sites



1 Request

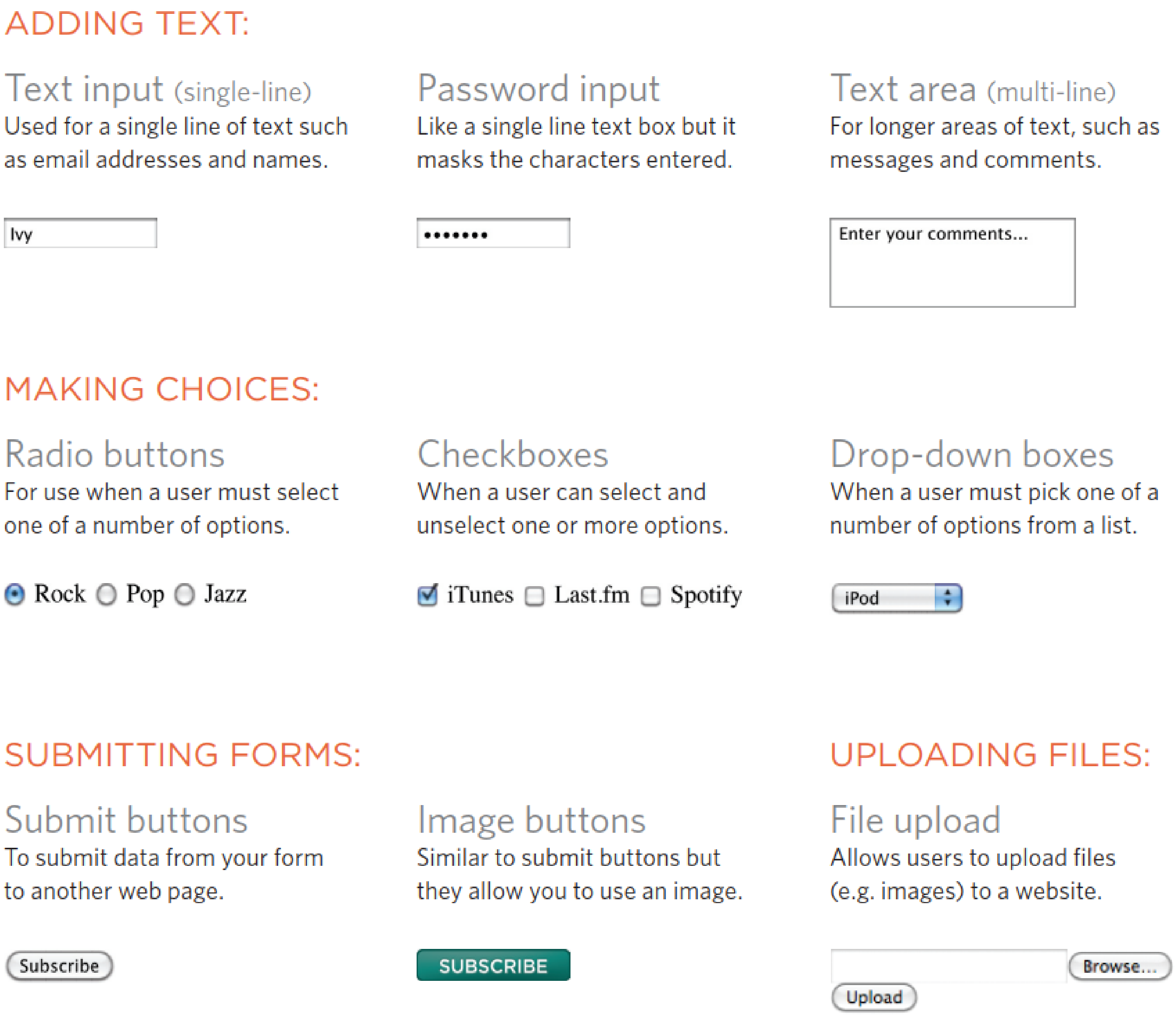
2 Browser returns HTML document that contains a form

3 User fills in form and submits the form

4 Request

The user's form data is sent to the server within the request.

5 This request is usually for some type of server-side script that will process the form data.



ADDING TEXT:

Text input (single-line)
Used for a single line of text such as email addresses and names.

Ivy

Password input
Like a single line text box but it masks the characters entered.

•••••••

Text area (multi-line)
For longer areas of text, such as messages and comments.

Enter your comments...

MAKING CHOICES:

Radio buttons
For use when a user must select one of a number of options.

● Rock ○ Pop ○ Jazz

Checkboxes
When a user can select and unselect one or more options.

☑ iTunes □ Last.fm □ Spotify

Drop-down boxes
When a user must pick one of a number of options from a list.

iPod

SUBMITTING FORMS:

Submit buttons
To submit data from your form to another web page.

Image buttons
Similar to submit buttons but they allow you to use an image.

UPLOADING FILES:

File upload
Allows users to upload files (e.g. images) to a website.

Forms provide rich mechanisms

There are several types of form controls

Type	Description
<form>	Defines the form container.
<input>	Defines an input field. **HTML5** defines over 20 different types of input.
<textarea>	Defines a multiline text entry box.
<button>	Defines a clickable button.
<label>	Defines a label for a form input element.
<datalist>	An **HTML5** element defines lists to be used with other form elements.
<fieldset>	Groups related elements in a form together.
<legend>	Defines the label for a fieldset group.
<select>	Defines a multi-item list.
<option>	Defines an option in a multi-item list.
<optgroup>	Defines a group of related options in a multi-item list.

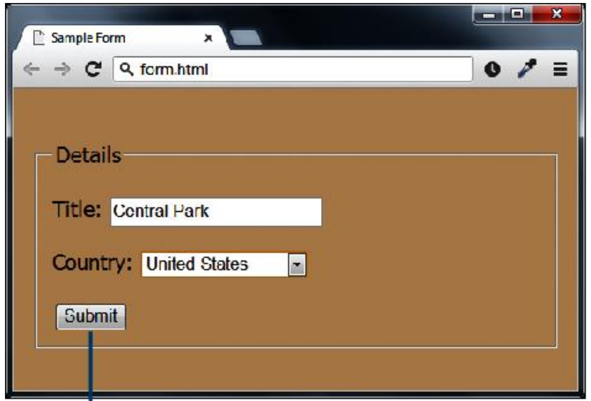
Form-Related HTML Elements

`<form> element`

- Form controls **live inside** a <form> element.
- Two essential attributes of any form element:
  - The action attribute
    - **Specifies the URL of the server-side** resource that will process the form data when it is submitted.
  - The method attribute
    - **Specifies how** the query string data will be transmitted from the browser to the server:
    - Either GET or POST

GET vs POST

The data is appended to the URL as a series of name/value pairs. Start with a question mark (?) followed by the name/value pairs, each one separated by an ampersand (&).



Details

Title: Central Park

Country: United States

```
<form method="get" action="process.php">
```

```
GET /process.php?title=Central+Park&where=United+States http/1.1
```

querystring

```
<form method="post" action="process.php">
```

```
POST /process.php http/1.1
Date: Sun, 20 May 2012 23:59:59 GMT
Host: www.mysite.com
User-Agent: Mozilla/4.0
Content-Length: 47

title=Central+Park&where=United+States
```

HTTP Header

querystring

The data is appended to the body of the HTTP request.

GET vs POST

GET

- Ideal for short forms
  - Such as search box or retrieving data from server
- Limit on the number of characters in the form data.
- Data can be clearly seen in the address bar.
- Data remains in browser history and cache.

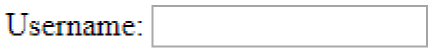
POST

- Ideal for
  - Data is very long
  - Data can contain binary data, e.g., uploading file.
  - Data contains sensitive data, e.g., password
- Submitted data is not stored in cache, history, or bookmarks.

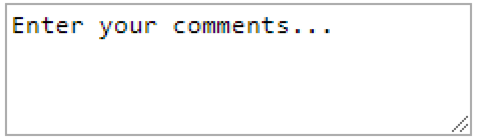
Input Controls

Type	Description
text	Creates a single line text entry box. `<p>Username: <input type="text" name="username" size="15" maxlength="30"></p>`
textarea	Creates a multiline text entry box. `<textarea name="comments" cols="30" rows="4">Enter your comments...</textarea>`
password	Creates a single line text entry box for a password. `<p>Password: <input type="password" name="password" size="15" maxlength="30"></p>`
hidden	Add a hidden data item to the form that users cannot see. `<input type="hidden" name="page" value="34">`

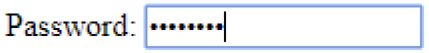
radio	Creates radio buttons for user to pick just one of the options.
	`<p>Please select your favorite genres: <br>`<br>`  <input type="radio" name="genre" value="action">Action`
	`  <input type="radio" name="genre" value="scific" checked>Science Fiction`
	`  <input type="radio" name="genre" value="drama">Drama`
	`</p>`



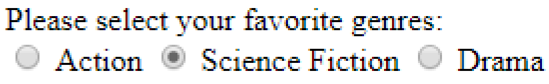
Username:



Enter your comments...



Password: ••••••••



Please select your favorite genres:
○ Action ● Science Fiction ○ Drama

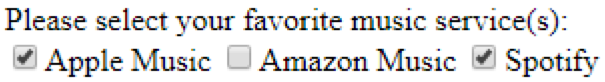
Input Controls

Type	Description

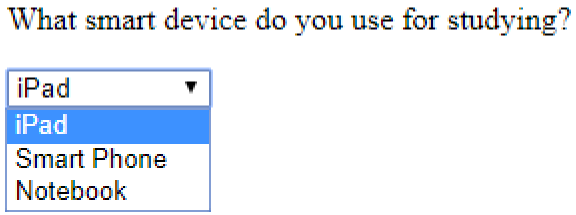
checkbox	Allow users to select (and unselect) one or more options in answer to a question.
	`<p>Please select your favorite music service(s):<br>`<br>`  <input type="checkbox" name="service" value="apple" checked>Apple Music`
	`  <input type="checkbox" name="service" value="amazon">Amazon Music`
	`  <input type="checkbox" name="service" value="spotify">Spotify`
	`</p>`

select	Creates a drop down list box for users to select one option from the list.
	`<p>What smart device do you use for studying?</p>`
	`<select name="devices">`
	`  <option value="ipad">iPad</option>`
	`  <option value="mobile">Smart Phone</option>`
	`  <option value="nbook">Notebook</option>`
	`</select>`

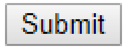
submit	For sending the form data to the server. `<input type="submit" value="Submit">`
reset	Defines a reset button that will reset all form values to their default values. `<input type="reset">`



Please select your favorite music service(s):
☑ Apple Music □ Amazon Music ☑ Spotify



What smart device do you use for studying?



Input Controls (HTML5 Elements)

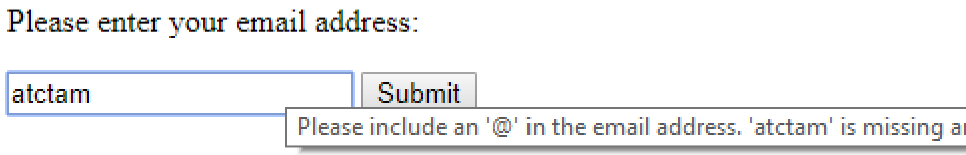
Type	Description

email	Creates a single-line text entry box suitable for entering an email address.
	`<p>Please enter your email address:</p>`
	`<input type="email" name="email">`
	`<input type="submit" value="Submit">`

url	Creates a single-line text entry box suitable for entering a URL.
	`<p>Please enter your website address:</p>`
	`<input type="url" name="website">`
	`<input type="submit" value="Submit">`

date	Creates a date input for entering a date.
	`<p>Departure date:`
	`  <input type="date" name="depart">`
	`  <input type="submit" value="Submit">`
	`</p>`

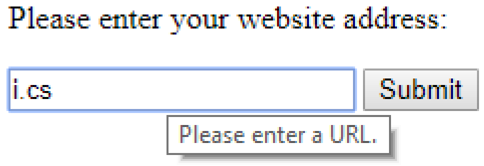
search	Creates a single-line text entry box suitable for a search string.
	`<p>Google Search:`
	`  <input type="search" name="search">`
	`  <input type="submit" value="Search">`
	`</p>`



Please enter your email address:

atctam

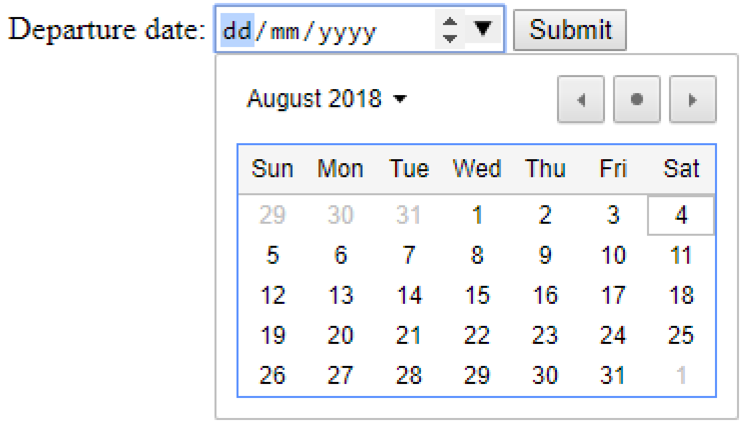
Please include an '@' in the email address. 'atctam' is missing a



Please enter your website address:

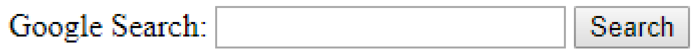
i.cs

Please enter a URL.



Departure date: dd/mm/yyyy

Sun	Mon	Tue	Wed	Thu	Fri	Sat
29	30	31	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1



Google Search:

More HTML5 Input Controls

- HTML5 added several new input types:
  - color
  - datetime-local
  - month
  - number
  - range
  - tel
  - time
  - week

Refer to “https://www.w3schools.com/html/html_form_input_types.asp” for more information

Character Entities

- These are **special characters** for symbols for which there is either no way easy way to type in via a keyboard (such as the copyright symbol © or accented characters) or which have a reserved meaning in HTML (for instance the "<" or ">" symbols).
- They can be used in an HTML document by using the entity name or the entity number.
- e.g., &nbsp; is &#160; and &copy; is &#169;

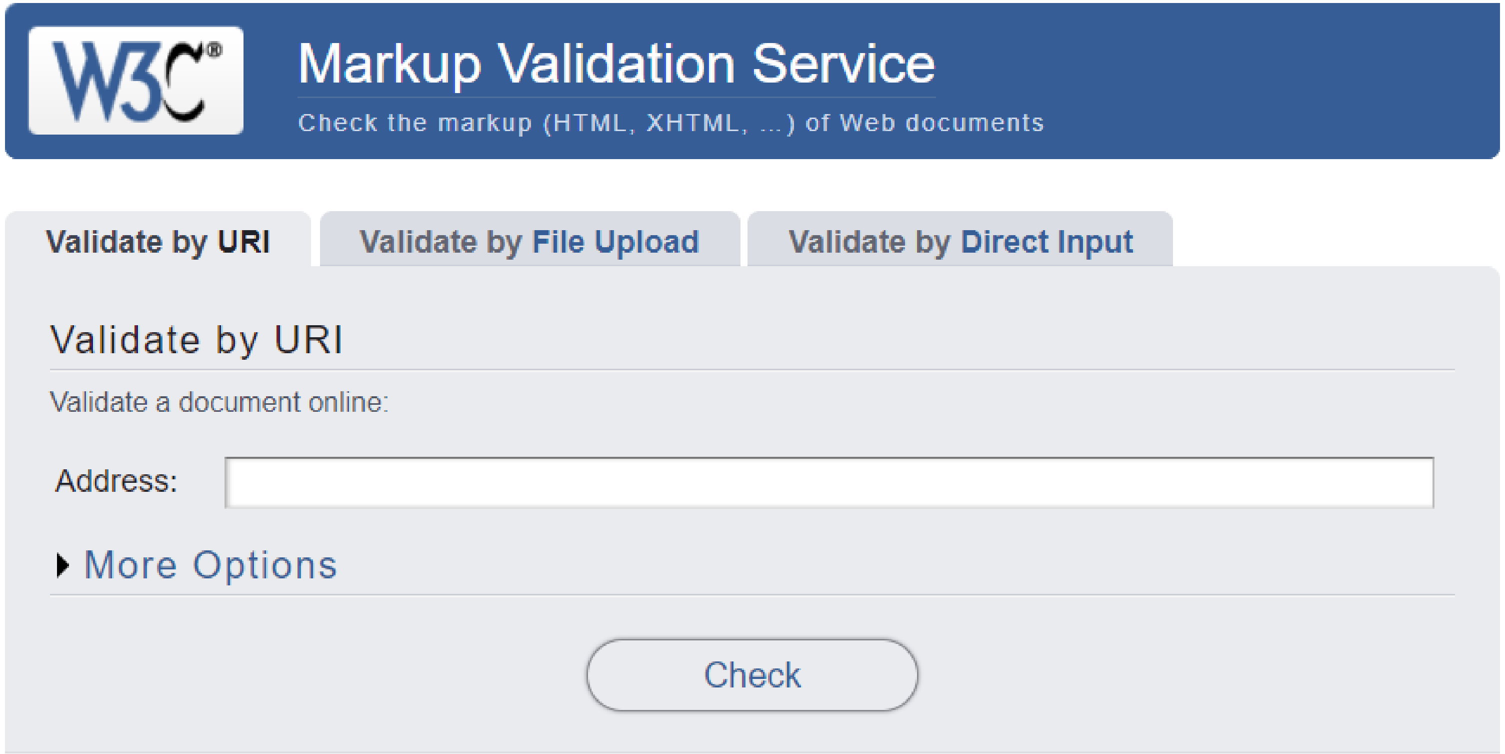
https://www.w3schools.com/charsets/ref_utf_latin1_supplement.asp

HTML Validation

- Coding HTML is programming, errors may exist in the HTML code
- HTML itself doesn't suffer from syntax errors because browsers parse it **permissively**
  - The browser still displays the page even if there are syntax errors.
  - Browser has built-in rules to interpret incorrectly written markup
    - Well, browser still displays the page but may not appear as what you expected
- How to make sure the HTML code is well-formed?
  - Use the Markup Validation Service to check your HTML code
  - https://validator.w3.org/ - created and maintained by the W3C

HTML Validation

- This webpage takes an HTML document as an input, goes through it, and gives you a report to tell you what is wrong with your HTML.



Markup Validation Service

Check the markup (HTML, XHTML, …) of Web documents

Validate by URI

Validate a document online:

Address:

HTML5 SEMANTIC ELEMENTS

HTML5 Semantic Elements

- One substantial problem with modern, pre-HTML5 semantic markup:
    - Most complex web sites are absolutely packed solid with <div> elements.
        - Unfortunately, all these <div> elements can make the resulting markup **confusing and hard to modify**.
    - Developers typically try to bring some sense and order to the <div> chaos by using **class or id attributes** to indicate the role of the <div> element in the structure of the page

HTML5 Semantic Elements

- The point of creating these new elements is that web page authors can use them to **describe the structure** of the page and the **meaning** of page content.
- Also, it is much easier to read and understand the code by the developers.
- These semantic structural elements make it easier for users to navigate the page using assistive technology for **accessibility**. For examples:
  - Screen reader software might allow users to ignore headers and footers and get straight to the content and read the content out loud for blind users.
  - Screen reader can recognize those elements and help with tasks like “find the main navigation”.

Web Accessibility

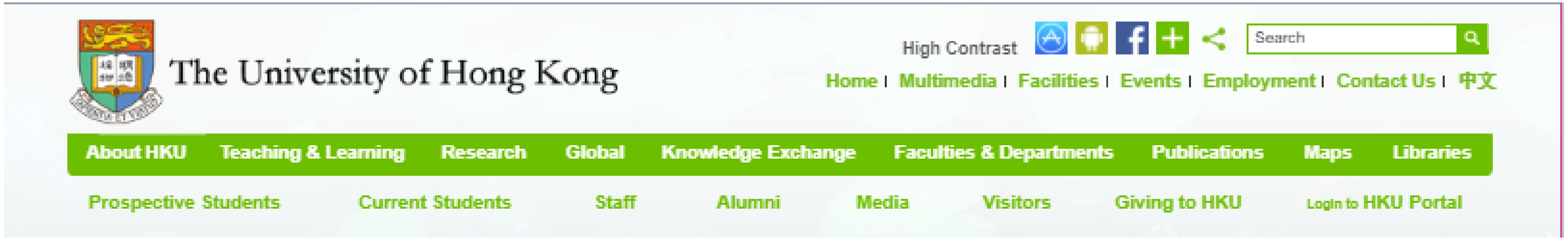
- A great deal of web content can be made "accessible" just by making sure the **correct HTML elements** are used for the correct purpose at all times.
- Semantic HTML doesn't take longer to write than non-semantic (bad) markup.
- Actually, it is easier to develop with semantic HTML as you may get some functionality for free and is easier to understand.
  - E.g., <button>Play video</button>
  - Browsers have some suitable styling applied by default and they may have **built-in keyboard accessibility**.

HTML5 Semantic Elements

- <footer>
- <header>
- <main>
- <nav>
- <article>
- <section>
- <aside>
- <figcaption>
- <figure>
- <details>
- <summary>
- <mark>
- <time>

Header <header> and Footer <footer>

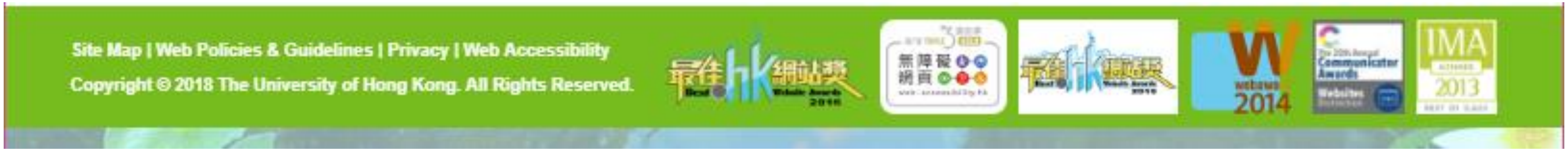
- Most web site pages have a recognizable header and footer sections.



- Header represents a group of introductory content. Typically the header contains
  - the site logo
  - title (and perhaps additional subtitles or taglines)
  - horizontal navigation links, and
  - perhaps one or two horizontal banners.

Header and Footer

- Footer represents a group of end content for a page.



Site Map | Web Policies & Guidelines | Privacy | Web Accessibility

Copyright © 2018 The University of Hong Kong. All Rights Reserved.

- Typically, the footer contains "less important" material, such as
  - smaller text versions of the navigation,
  - copyright notices,
  - information about the site's privacy policy, and
  - perhaps twitter feeds or links to other social sites.

Header and Footer

- Both the HTML5 <header> and <footer> element can be used not only for *page* headers and footers, they can also be used for header and footer elements within other HTML5 containers, such as <article> or <section>.

```
<header>
   <img src=“icon.gif" alt=“icon">
   <h1> Principles of Web Development</h1>
   ...
</header>
<article>
   <header>
      <h2>HTML5 Semantic Elements </h2>
      <p>By <em>COMP3322</em></p>
      <p><time>January 1, 2020</time></p>
   </header>
   ...
</article>
```

Main Content <main>

- The <main> tag specifies the main content of a document, which **should be unique** to that page.
  - Use <main> only once per page.
  - It **should not contain** any content that is repeated across documents such as sidebars, navigation links, copyright information, site logos, and search forms.
  - **May contain** subsections represented by <article>, <section>, and <div> elements.

Navigation <nav>

- The <nav> element represents a section of a page that contains links to other pages or to other parts within the same page.
  - The <nav> element was intended to be used for major navigation blocks. Secondary links, etc., would not go in the navigation.

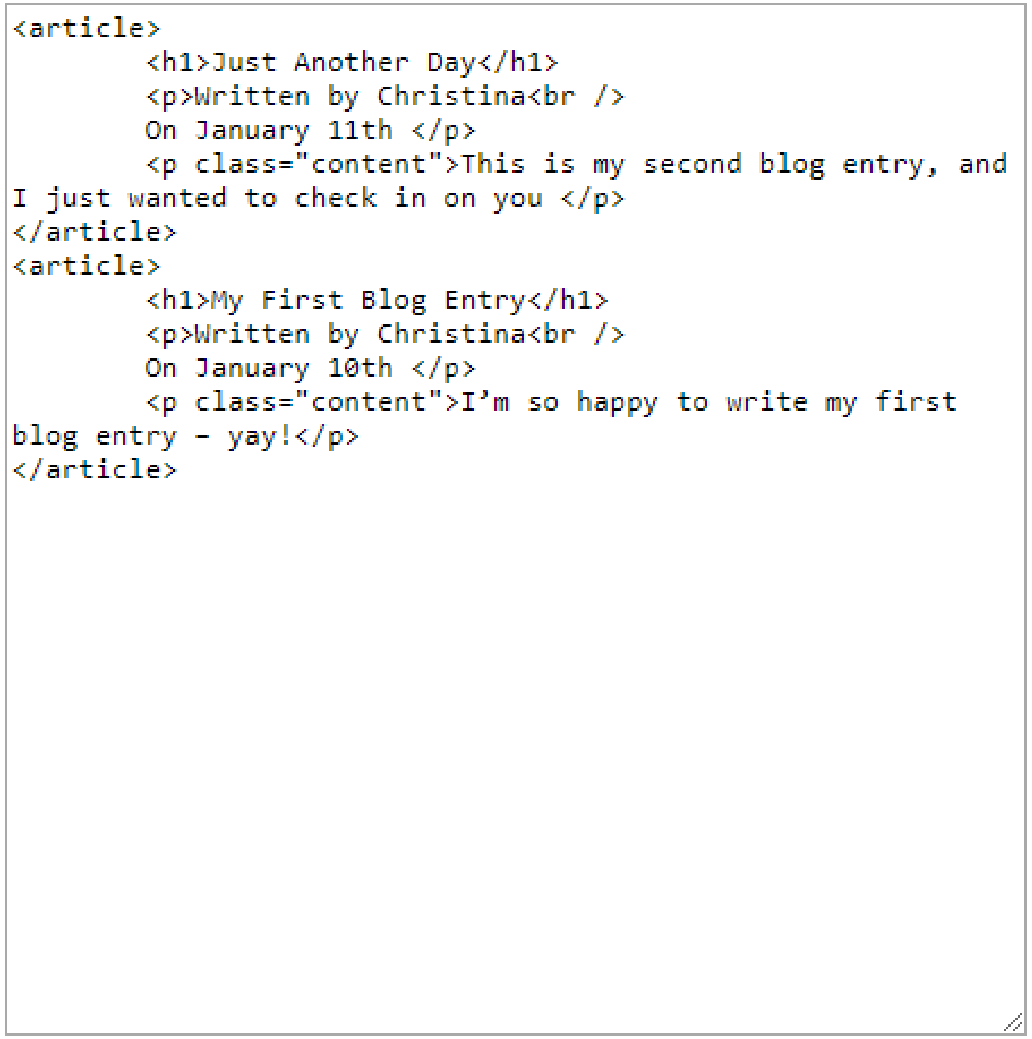
```
<header>
   <img src=“icon.gif" alt=“icon" />
   <h1>Principles of Web Development</h1>
   <nav role="navigation">
      <ul>
         <li><a href="index.html">Home</a></li>
         <li><a href="about.html">About Us</a></li>
         <li><a href="contact.html">Contact Us</a></li>
      </ul>
   </nav>
</header>
```

https://developer.mozilla.org/en-US/docs/Web/HTML/Element/article

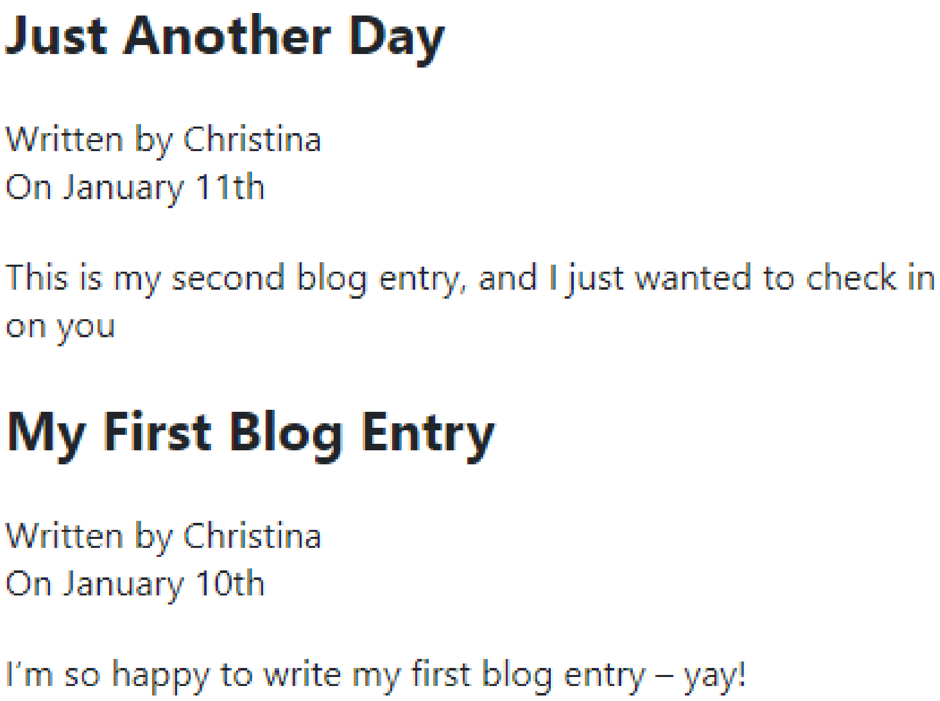
Articles <article>

- The <article> element represents **a section** of **content** that
  - is **semantically related**, should also have a heading, and should be able to be isolated from the rest of the page and still be meaningful.
  - For example, a magazine or newspaper article, or a blog entry.
- A given document can have multiple articles in it.
  - For example, on a blog that shows the text of each article one after another; each post would be contained in an <article> element, possibly with **one or more <section>s** within.

Articles <article>



```
<article>
        <h1>Just Another Day</h1>
        <p>Written by Christina<br />
        On January 11th </p>
        <p class="content">This is my second blog entry, and
I just wanted to check in on you </p>
</article>
<article>
        <h1>My First Blog Entry</h1>
        <p>Written by Christina<br />
        On January 10th </p>
        <p class="content">I’m so happy to write my first
blog entry – yay!</p>
</article>
```



Just Another Day

Written by Christina
On January 11th

This is my second blog entry, and I just wanted to check in on you

My First Blog Entry

Written by Christina
On January 10th

I’m so happy to write my first blog entry – yay!

https://developer.mozilla.org/en-US/docs/Web/HTML/Element/section

Sections <section>

- The <section> element represents a **generic** section of content that
  - ***can* be grouped together** in a semantically meaningful way;
  - they should have a “theme”
    - A <section>'s "theme" should be defined by including a heading element within the element, often immediately after the opening tag.

- Examples of sections would be chapters, the numbered sections of a document, news items, etc.
- According to the W3C, <section> is a **much broader element**, while the <article> element is to be used for blocks of content that could potentially be read or consumed independently of the other content on the page.

Sections versus Divs

- The WHATWG specification warns readers that the <section> element is not a generic container element. HTML already has the <div> element for such uses.
- When an element is needed only for styling purposes or as a convenience for scripting, it makes sense to use the <div> element instead.
- Another way to help you decide whether or not to use the <section> element is to ask yourself if it is appropriate for the element's contents to be **listed explicitly in the document's outline**.
- If so, then use a <section>; otherwise use a <div>.

Figure and Figure Captions

- The <figure> element should not be used to **wrap every image**.
  - For instance, it makes no sense to wrap the site logo or non-essential images such as banner ads and graphical embellishments within <figure> elements.
- Instead, only use the <figure> element for circumstances where the image (or other content) has a caption and where the figure is **essential** to the content but its position on the page is **relatively unimportant**.

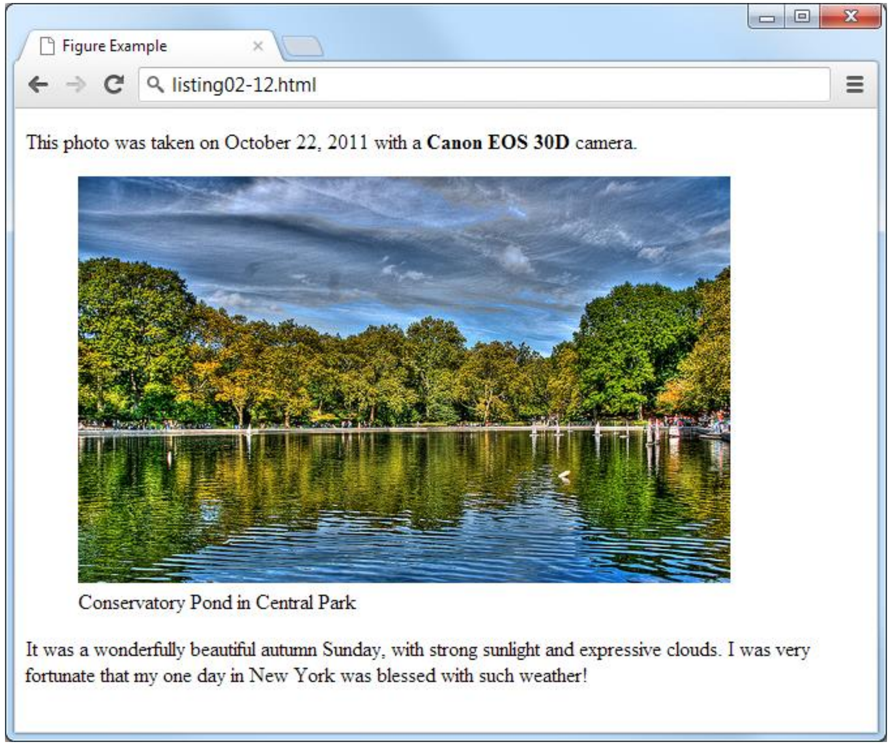
Figure and Figure Captions

Figure could be moved to a different location in document

…

But it has to exist in the document (i.e., the figure isn’t optional)

```
<p>This photo was taken on October 22, 2011 with a Canon EOS 30D camera.</p>
<figure>
  <img src="images/central-park.jpg" alt="Central Park" /><br/>
  <figcaption>Conservatory Pond in Central Park</figcaption>
</figure>
<p>
It was a wonderfully beautiful autumn Sunday, with strong sunlight and
expressive clouds. I was very fortunate that my one day in New York was
blessed with such weather!
</p>
```



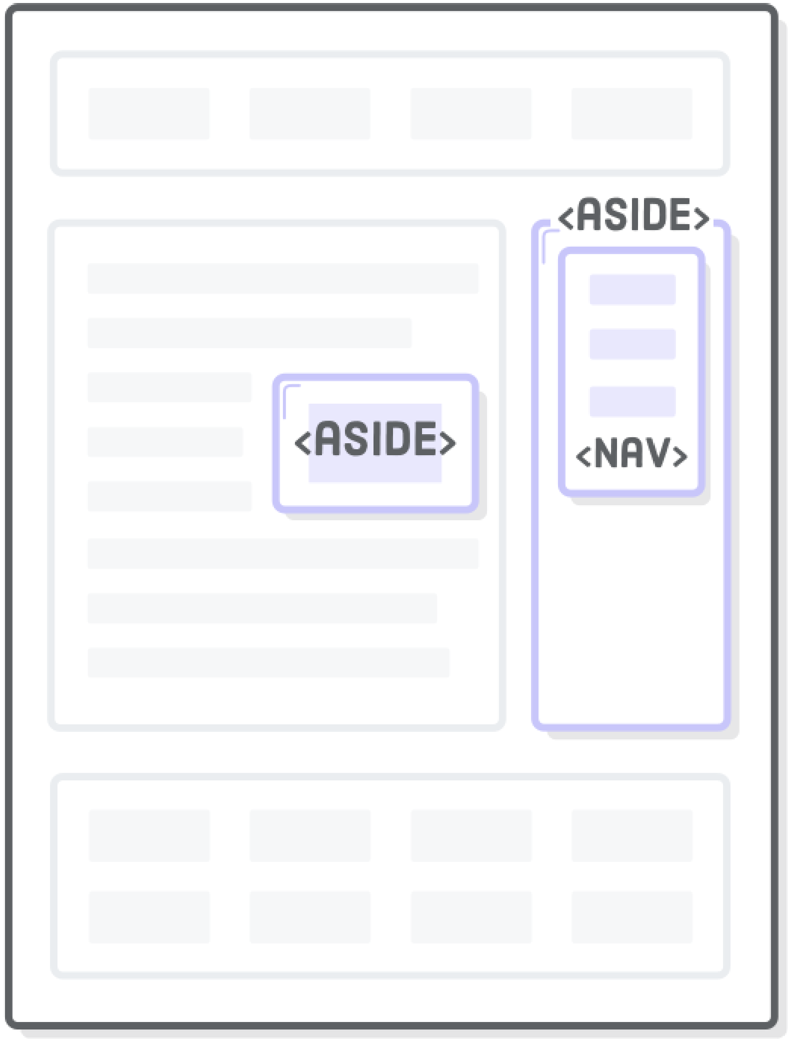
This photo was taken on October 22, 2011 with a **Canon EOS 30D** camera.

Conservatory Pond in Central Park

It was a wonderfully beautiful autumn Sunday, with strong sunlight and expressive clouds. I was very fortunate that my one day in New York was blessed with such weather!

Aside

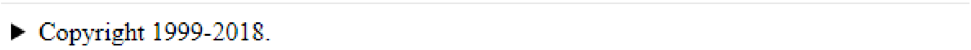
- The <aside> element contains content that is **not part of the flow of the text** in which it appears, however still related to the main content in some way.
  - It can **provide additional information** indirectly related to the main content (e.g., author biography, related links, etc.)
- So <aside> is similar to the <figure> element in that it is used for marking up content that is separate from the main content on the page.
- The <aside> element could thus be used for sidebars, call-out boxes, groups of advertising images, or any other grouping of non-essential elements.



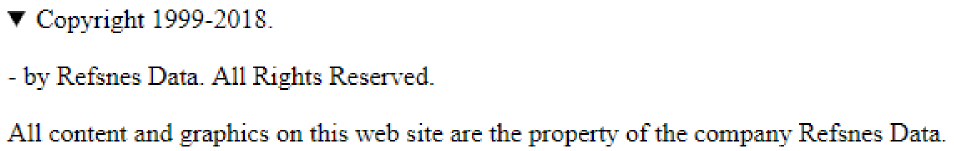
<details> <summary>

- The <details> tag can be used to create an interactive widget that the user can open and close.
- Any sort of content can be put inside the <details> tag.
- The <summary> tag defines a visible heading for the <details> element. The heading can be clicked to view/hide the details.

```
<details>
  <summary>Copyright 1999-2018.</summary>
  <p> - by Refsnes Data. All Rights
Reserved.</p>
  <p>All content and graphics on this web
site are the property of the company
Refsnes Data.</p>
</details>
```



▶ Copyright 1999-2018.



▼ Copyright 1999-2018.

- by Refsnes Data. All Rights Reserved.

All content and graphics on this web site are the property of the company Refsnes Data.

<time> <mark>

- The <time> element defines a human-readable date/time.
  - It may include the *datetime attribute* to translate dates into machine-readable format so that user agents can offer to add birthday reminders or scheduled events to the user's calendar, and search engines can produce smarter search results.

```
<p>I have a date on <time datetime="2020-02-14 20:00">Valentines day</time>.</p>
```

- Use the <mark> element if you want to mark or highlight text for reference or notation purposes.

```
<p>Do not forget to buy <mark>milk</mark> today.</p>
```

Reading

- MDN web docs
  - Introduction to HTML
    - https://developer.mozilla.org/en-US/docs/Learn/HTML/Introduction_to_HTML

References

- Some slides are borrowed from the book:
  - Fundamentals of Web Development by Randy Connolly and Ricardo Hoar, published by Pearson.

- Semantic HTML, No. 12 of HTML & CSS is Hard
  - https://internetingishard.com/html-and-css/semantic-html/