HTML Part 2

ITCS 210 Web Programming with Activities

Instructor: Pawitra Chiravirakul
Email: pawitra.chi@mahidol.ac.th
Section 1 : Friday 7th September 2018, 1.00-4.00p.m., IT106
Section 2 : Monday 3rd September 2018, 1.00-4.00p.m., IT106
Section 3 : Thursday 6th September 2018, 9.00-12.00p.m., IT106

Class Objectives

- To use HTML and HTML5 to create web pages.
- To learn the basic structure, and how they are used, including creating form validation to check the value in a form, inserting video, audio, creating and use forms to get user input.

Forms

- A form on a web page allows a user to enter information and submit it for processing.
- A form is defined by <form> </form> tags and contains form elements.
- Form elements (like text fields, text area, drop-down menus, radio buttons, checkboxes, etc.) are elements that allow the user to enter information into a form.
- Attribute **`method`** specifies how the form's data is sent to the web server
- The **`action`** attribute of the form element specifies the script to which the form data will be sent

Form example

First name:
Last name:
Gender: Male Female
Speaking language: Thai English Chinease French

```
<form>
First name: <input type="text" name="firstname"><br>
Last name: <input type="text" name="lastname"><br>
Gender: <input type="radio" name="sex" value="male" checked> Male
    <input type="radio" name="sex" value="female"> Female<br />
Speaking language:
<input type="checkbox" name= "language" value= "Thai">Thai
<input type="checkbox" name="language " value= "English">English
<input type="checkbox" name="language " value= "Chinese">Chinese
<input type="checkbox" name="language " value= “French">French<br>

</form>
```

Forms: Input Tags

- The most used form element is the **<input>** tag.
- The type of input is specified with the **`type`** attribute.
- Input types can be:
  - Text Fields, Password Fields, Radio Buttons, Checkboxes, Selection Menus, Reset and Submit Buttons, Text area

Text Fields

- Text fields are used when you want a user to type letters, numbers, etc. in a form.

```
First name: <input type="text" name="firstname">
<br>
Last name: <input type="text" name="lastname">
```

- How it looks in a browser:

Password Fields

- Password fields are like text fields, except that the text entered is not shown.

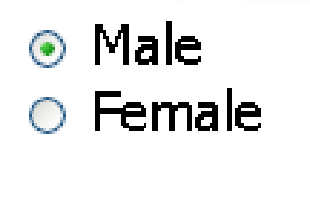
```
Login: <input type="text" name="login">
<br>
Password: <input type="password" name="password">
```

- How it looks in a browser:

Radio Buttons

- Radio Buttons are used when you want a user to select one of a limited number of choices.

```
<input type="radio" name="sex" value="male" checked> Male
<br>
<input type="radio" name="sex" value="female"> Female
```



Must be the same name

- How it looks in a browser:

- Note that **<u>only one option can be chosen</u>**.

Checkboxes

- Checkboxes are used when you want a user to select one or more options of a limited number of choices.

```
I have a bike:
<input type="checkbox" name="vehicle" value="Bike">
<br />
I have a car:
<input type="checkbox" name="vehicle" value="Car">
<br />
I have an airplane:
<input type="checkbox" name="vehicle" value="Airplane">
```

When you have several checkboxes With the *same name*, make sure that they have *different values*

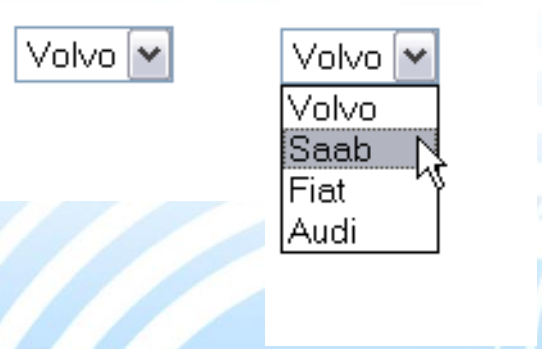
I have a bike:
I have a car:
I have an airplane:

- How it looks in a browser:
- Note that **<u>many options can be chosen</u>**.

Selection Menus (Drop down list)

- A selection menus presents a list of options.

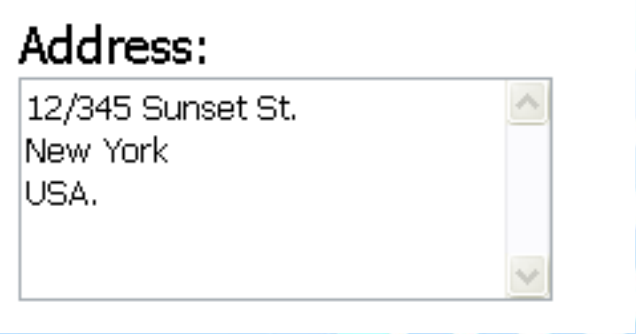
```
<select name="cars">
    <option value="volvo">Volvo</option>
    <option value="saab">Saab</option>
    <option value="fiat">Fiat</option>
    <option value="audi">Audi</option>
</select>
```



Textarea

- Textareas are multi-line text input.

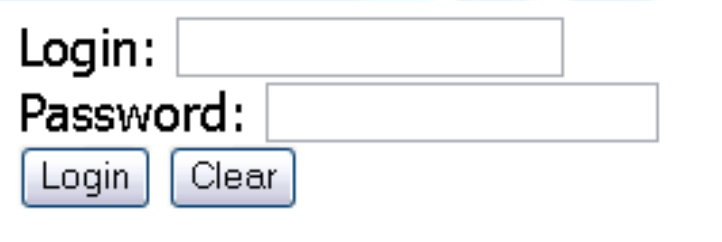
```
Address: <br>
<textarea name="address" rows="5" cols="30">
</textarea>
```



Reset and Submit Buttons

- A reset button changes a form back to its original state.
- A submit button sends user inputs to a server for processing.

```
Login: <input type="text" name="login">
<br>
Password: <input type="password" name="password">
<br>
<input type="submit" value="Login"> <input type="reset" value="Clear">
```



Action Attribute and the Submit Button

- When a user clicks on the "Submit" button, the content of the form is sent to another file. The form's action attribute defines the name of the file to send the content to. The file defined in the action attribute usually does something with the received input.

```
<form name="input" action="html_form_action.php"  method="get">
Username:
    <input type="text" name="user">
    <input type="submit" value="Submit" style="font-weight: bold">
</form>
```

This page will be run after click the submit button

Two available methods:

1) **get** – Data are submitted via the URL
2) **post** – Data are submitted inside the packet

Action Attribute and the Submit Button

- How it looks in a browser:

Username: Submit

- If you type some characters in the text field above, and click the "Submit" button,
  - You will send your input to a page called "html_form_action.php"
  - That page will show you the received input.

Field Set

- Used to draw a border with a caption around your data

```
<fieldset>
<legend>
Registration Form
</legend>
<form action="">
Firstname : <input type="text" size="10"><br><br>
Lastname : <input type="text" size="10"><br><br>
Age : <input type="text" size="2"><br><br>
Marriage Status : Single
<input type="radio" name="status" value="single"
checked="checked">
Marrie
<input type="radio" name="status"
value="marrie">
</form>
</fieldset>
```

What is HTML 5

- HTML 5 will be the new standard for HTML
- HTML 5 is the next major version of HTML
- It introduces a bunch of new elements that will make our pages more semantic
- It can help you design better-looking and more interactive sites.
- It supports video and audio, making the use of Flash obsolete.
- It offers geolocation support, offline storage, better forms and better support for web apps.

New HTML5 elements

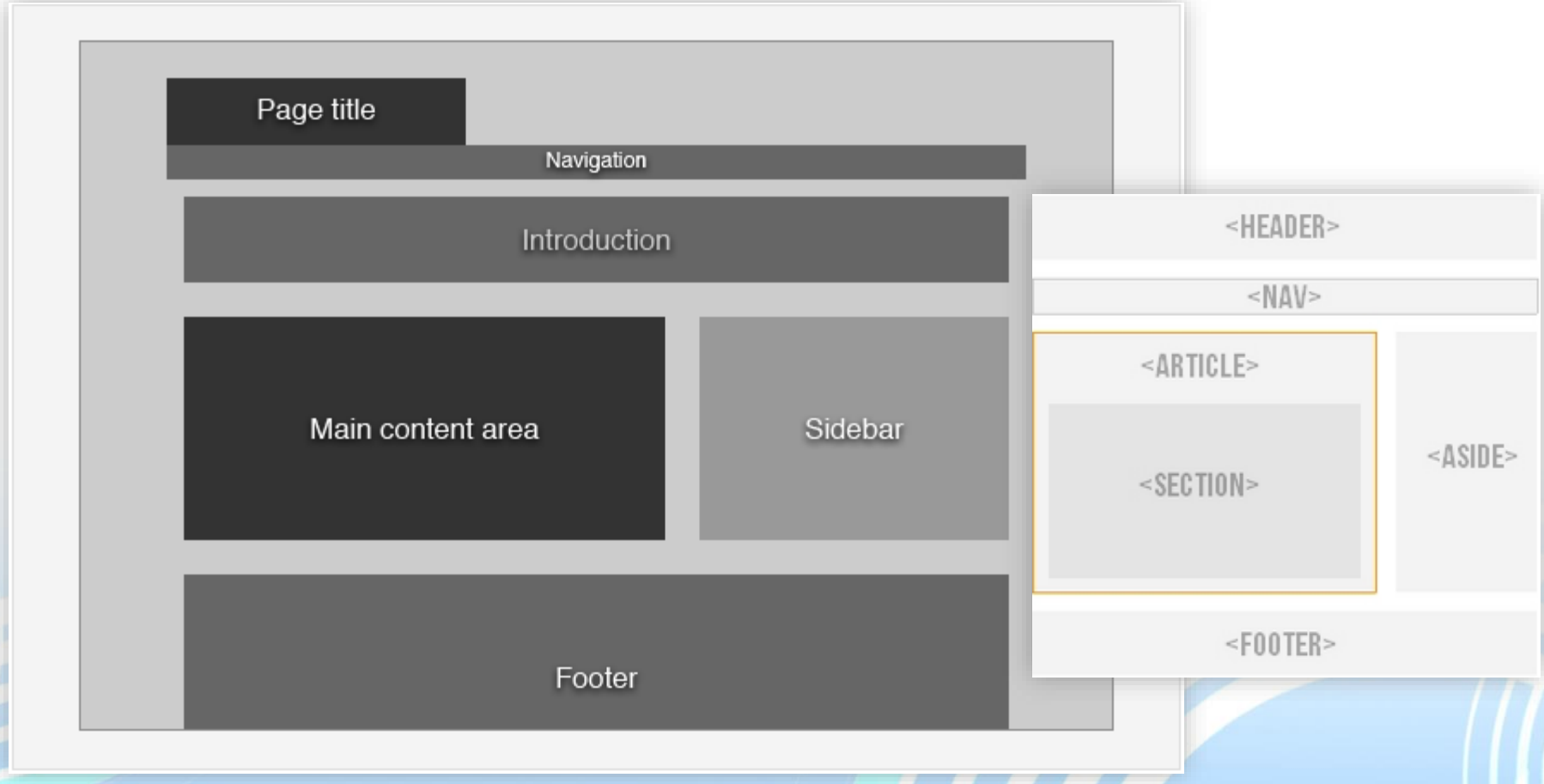
- New **semantic elements** like <header>, <footer>, <nav>, <article>, and <section>.
- New **attributes of form elements** like number, date, time, calendar, and range.
- New **multimedia elements**: <audio> and <video>
- New **graphic elements**: <svg> and <canvas>.

Migration from HTML4 to HTML5

Typical HTML4	Typical HTML5
	<header>
	<nav>
	<section>
	<article>
	<footer>

https://www.tutorialrepublic.com/html-reference/html5-tags.php

Basic Structure of HTML5



Removed Elements in HTML5

Removed Element	Use Instead
<acronym>	<abbr>
<applet>	<object>
<basefont>	CSS
<big>	CSS
	CSS
<dir>	
<font>	CSS

Basic Structure of HTML5

- <header></header>
  - Represents a group of introductory or navigational aids.

```
<header>
        <p>Welcome to...</p>
        <h1>Voidwars!</h1>
</header>
```

Basic Structure of HTML5

```
<body>
        <header>
                <h1>Little Green Guys With Guns</h1>
                <nav>
                        <ul>
                                <li><a href="/games">Games</a>
                                <li><a href="/forum">Forum</a>
                                <li><a href="/download">Download</a>
                        </ul>
                </nav>
                <h2>Important News</h2>
                <p>To play today's games you will need to update your client.</p>
                 <h2>Games</h2>
        </header>
        …
</body>
```

Basic Structure of HTML5

- <hgroup></hgroup>
  - Represents a group of introductory or navigational aids.
  - The element is used to group a set of h1–h6 elements when the heading has multiple levels

```
<hgroup>
        <h1>The reality dysfunction</h1>
        <h2>Space is not the only void</h2>
 </hgroup>

<hgroup>
        <h1>Dr. Strangelove</h1>
        <h2>Or: How I Learned to Stop Worrying and Love the Bomb</h2>
</hgroup>
```

Basic Structure of HTML5

- <nav></nav>
  - represents a section of the document intended for navigation.

```
<body>
        <h1>The Wiki Center Of Exampland</h1>
        <nav>
                <ul>
                        <li><a href="/">Home</a></li>
                        <li><a href="/events">Current Events</a></li>
                        ...more...
                </ul>
        </nav>
        …more…
</body>
```

Basic Structure of HTML5

- <article></article>
  - Represents an independent piece of content of a document, such as a blog entry or newspaper article.
- <section></section>
  - Represents a generic document or application section.

Basic Structure of HTML5

```
<article>
    <hgroup>
        <h1>Apples</h1>
        <h2>Tasty, delicious fruit!</h2>
    </hgroup>
    <p>The apple is the pomaceous fruit of the apple tree.</p>
    <section>
        <h1>Red Delicious</h1>
        <p>These bright red apples are the most common found in many
        supermarkets.</p>
    </section>
    <section>
        <h1>Granny Smith</h1>
        <p>These juicy, green apples make a great filling for apple pies.</p>
    </section>
</article>
```

Basic Structure of HTML5

- <aside></aside>
  - Represents a piece of content that is only slightly related to the rest of the page.
- <footer></footer>
  - Represents a footer for a section and can contain information about the author, copyright information, etc.

Basic Structure of HTML5

```html
<!doctype html>
<html>
<head>
    <title>Page title</title>
</head>
<body>
    <header>
        <h1>Page title</h1>
    </header>
    <nav>
        <!-- Navigation -->
    </nav>
    <section id="intro">
        <!-- Introduction -->
    </section>
    <section>
        <!-- Main content area -->
    </section>
    <aside>
        <!-- Sidebar -->
    </aside>
    <footer>
        <!-- Footer -->
    </footer>

</body>
</html>
```

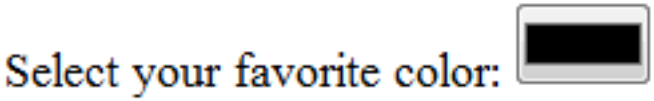
HTML5 Form Types

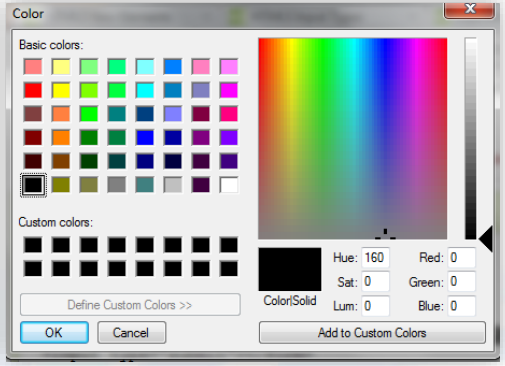
- The input element's type attribute now has the following new values:

  - tel
  - search
  - url
  - email
  - datetime
  - date
  - month
  - week
  - time
  - datetime-local
  - number
  - range
  - color

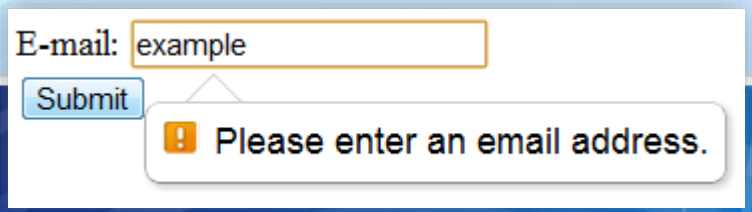
New Form Types

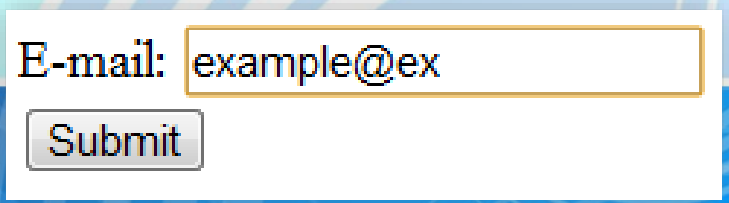
- Input Type: color
  - <input type="color" name="favcolor">





- Input Type: email
  - <input type="email" name="usermail">





New Form Types

- Input Type: number

  – <input type=”number" name="points" min="1" max="10">

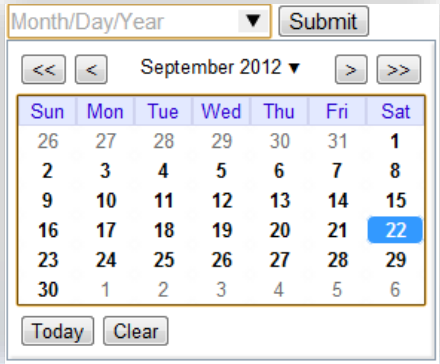
- Input Type: range
  – <input type="range" name="points" min="1" max="10">

- max - specifies the maximum value allowed
- min - specifies the minimum value allowed
- step - specifies the legal number intervals
- value - Specifies the default value

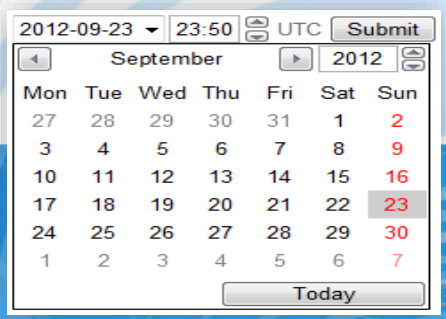
New Form Types

- Input Type: date
  - <input type="date" name="bday">

Birthday: Month/Day/Year ▼ Submit

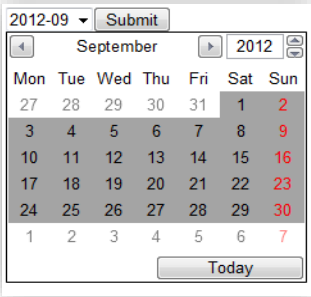


- Input Type: datetime
  - <input type="datetime" name="bdaytime">



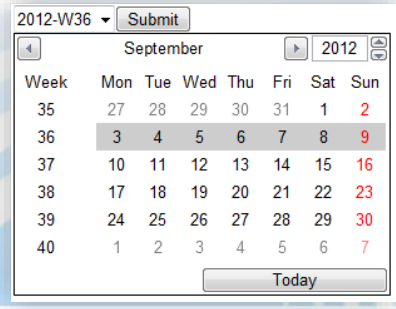
New Form Types

- Input Type: month
  - <input type="month" name="bdaymonth">



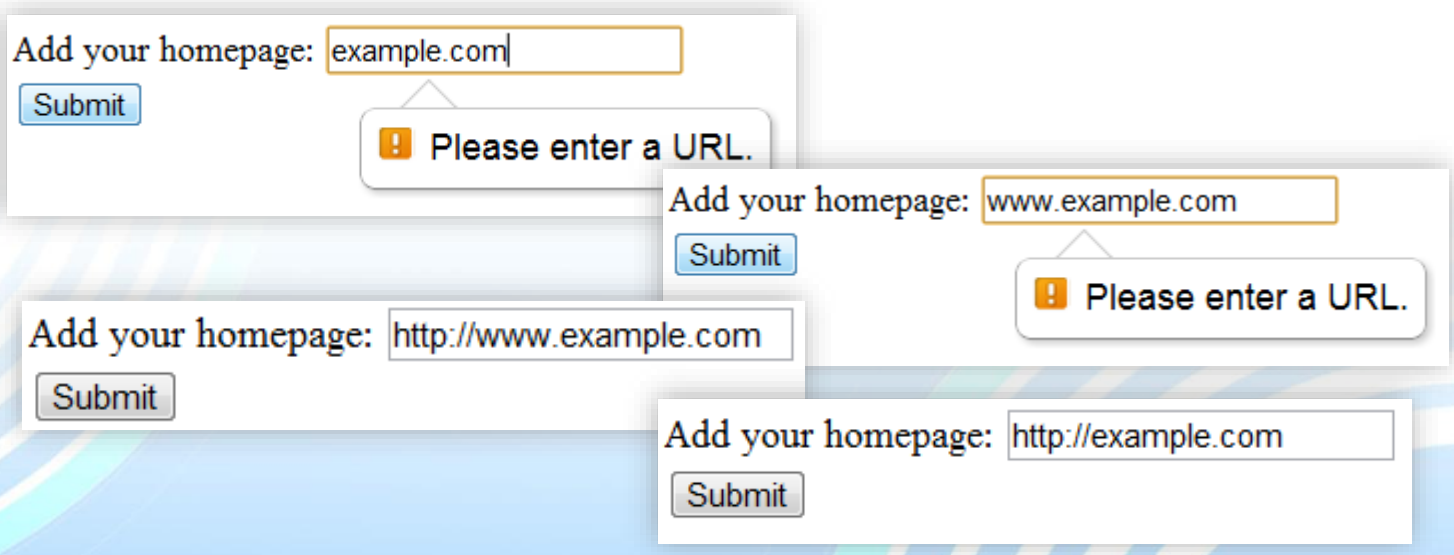
- Input Type: time
  - <input type="time" name="usr_time">

- Input Type: week
  - <input type="week" name="year_week">



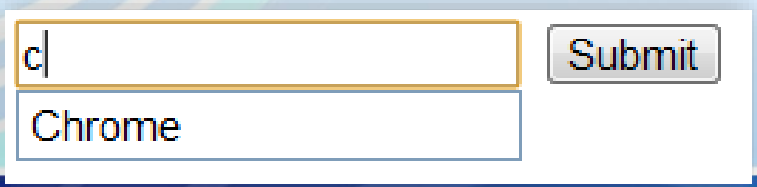
New Form Types

- Input Type: url
  - <input type="url" name="homepage">



New Form Elements

- Specifies a list of pre-defined options for an <input> element
  - It is used to provide an "autocomplete" feature on <input> elements. Users will see a drop-down list of pre-defined options as they input data.
  - The <datalist> tag can be used in conjunction with an <input> element that contains a list attribute.
  - The list attribute is linked to the <datalist> tag by the <datalist> tag's ID. For example, if the <datalist> tag contains id="browsers", then the list attribute will look like this: list="browsers".
  - You can fill the <datalist> element by nesting <option> tags inside the <datalist> tag.

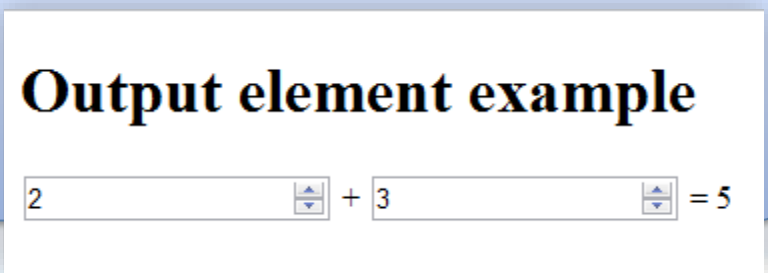



```
<input list="browsers" name="browser">
<datalist id="browsers">
  <option value="Internet Explorer">
  <option value="Firefox">
  <option value="Chrome">
  <option value="Opera">
  <option value="Safari">
</datalist>
```

New Form Element

- The output element represents the result of a calculation

```
<form onsubmit="return false" oninput="o.value =parseInt(a.value) + parseInt(b.value)">
  <input name="a" type="number" step="any"> +
  <input name="b" type="number" step="any"> =
  <output name="o"></output>
</form>
```




```
<form onsubmit="return false" oninput="o.value = a.valueAsNumber + b.valueAsNumber">
  <input name="a" type="number" step="any">
  <input name="b" type="number" step="any">
  <output name="o"></output>
</form>
```

returns the value as a number rather than as a string

New Form Attributes

- **New attributes for <form>:**
  - autocomplete
  - novalidate

- **New attributes for <input>:**
  - autocomplete
  - autofocus
  - form
  - formaction
  - formenctype
  - formmethod
  - formnovalidate
  - formtarget
  - height and width
  - list
  - min and max
  - multiple
  - pattern (regexp)
  - placeholder
  - required
  - step

New Form Attributes

- autocomplete attribute
  - The autocomplete attribute specifies whether a form or input field should have autocomplete on or off.
  - <form autocomplete="on"></form>
  - <input type="text" autocomplete="off" />
- novalidate attribute
  - It specifies that the form-data (input) should not be validated when submitted.
  - <form novalidate></form>

New Form Attributes

- form attribute
  - Allows you to associate any orphaned form control with any <form> element on the page

```
<form id="form1" method="get">
        First name: <input type="text" name="fname"><br>
        <input type="submit" value="Submit">
</form>
Last name: <input type="text" name="lname" form="form1">
```

- formnovalidate attribute
  - Specifies that the <input> element should not be validated when submitted

New Form Attributes

- formaction attribute
  - Specifies the URL of a file that will process the input control when the form is submitted
  - Overrides the action attribute of the <form> element
- formmethod attribute
  - Defines the HTTP method for sending form-data to the action URL
  - Some possible values are “get” or “post”
- formtarget attribute
  - Specifies a name or a keyword that indicates where to display the response that is received after submitting the form
  - Some possible values are “_blank”, “_self”, “_parent”, “_top”

- Override those attributes of the <form> element
- Used with type=“submit” and type=“image”

New Form Attributes

```
<html>
        <body>
                <form action= "demo.html" method="get">
                        First name: <input type="text" name="fname"><br>
                        Last name: <input type="text" name="lname"><br>
                        Email: <input type="email" name="emailInput"><br>
                        <input type="submit" value="Submit">
                        <input type="submit" formmethod="get"
                        formaction= "test.html"  formtarget="_blank"
                        formnovalidate value="Submit HTML5">
                </form>
        </body>
</html>
```

New Form Attributes

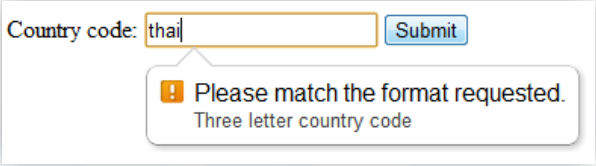
- autofocus attribute
  - Specifies that an <input> element should automatically get focus when the page loads
  - <input type="text" name="fname" autofocus>

New Form Attributes

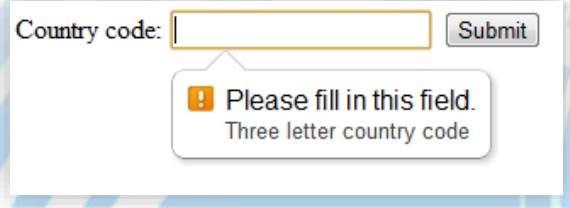
- min and max attributes
  - Specify the minimum and maximum value for an <input> element
  - Enter a date before 1980-01-01:
    <input type="date" name="bday" max="1979-12-31">
  - Enter a date after 2000-01-01:
    <input type="date" name="bday" min="2000-01-02">
  - Quantity (between 1 and 5):
    <input type="number" name="quantity" min="1" max="5">
- step attribute
  - Specifies the legal number intervals for an <input> element
  - <input type="number" name="quantity" min="1" max="5" step= "2" >

New Form Attributes

- pattern attribute
  - Specifies a regular expression that the <input> element's value is checked against
  - <input type="text" pattern="[A-Za-z]{3}">

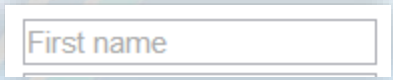


- require attribute
  - Specifies that an input field must be filled out before submitting the form
  - <input type="text" required>



New Form Attributes

- placeholder attribute
  - Specifies a short hint that describes the expected value of an input field
  - <input type="text" name="fname" placeholder="First name">



Video

```
<video width="320" height="240" controls>
        <source src="movie.mp4" type="video/mp4" />
        <source src="movie.ogg" type="video/ogg" />
        Your browser does not support the video tag.
</video>
```

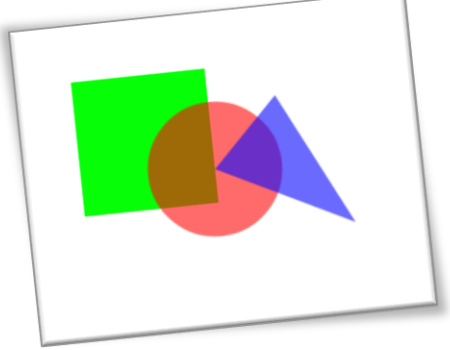
- Currently, there are 3 supported video formats for the <video> element: MP4, WebM, and Ogg
- autoplay attribute → autoplay
- loop attribute → loop

Audio

```
<audio controls>
        <source src="song.ogg" type="audio/ogg" />
        <source src="song.mp3" type="audio/mpeg" />
        Your browser does not support the audio element.
</audio>
```

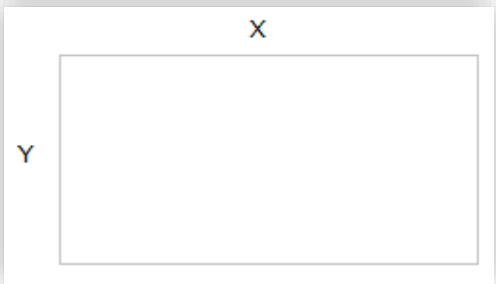
- Currently, there are 3 supported file formats for the <audio> element: MP3, Wav, and Ogg

Canvas



- The <canvas> element is used to draw graphics on a web page via scripting.
- The <canvas> element is only a container for graphics. You must use a script to actually draw the graphics.
- A canvas is a rectangular area on an HTML page
- Always specify an id attribute (to be referred to in a script), and a width and height attribute to define the size of the canvas.
- <canvas id="myCanvas" width="200" height="100"></canvas>

Canvas



- Add a border to a canvas box
  - <canvas id="myCanvas" width="200" height="100" style="border:1px solid #000000;"> </canvas>
- All drawing on the canvas must be done inside a JavaScript
- The canvas is a two-dimensional grid
- More Info:
  - http://www.w3schools.com/html5/html5_ref_canvas.asp

Canvas

- Draw rectangle

```
<script>
        var c=document.getElementById("myCanvas");
        var ctx=c.getContext("2d");
        ctx.fillStyle="#FF0000";
        ctx.fillRect(0,0,150,75);
</script>
```

- fillStyle
  - Sets or returns the color, gradient, or pattern used to fill the drawing
- fillRect(*x,y,width,height*)
  - draws a rectangle filled with the current fill style

Canvas

- Radial Gradients

```
<script>
	var c=document.getElementById("myCanvas");
	var ctx=c.getContext("2d");
	var grd=ctx.createRadialGradient(75,50,5,90,60,100);
	grd.addColorStop(0,"red");
	grd.addColorStop(1,"white");
	ctx.fillStyle=grd;
	ctx.fillRect(10,10,150,100);
</script>
```

Canvas

- Draw Text
- font - defines the font properties for text
- fillText(*text,x,y*) - Draws "filled" text on the canvas
- strokeText(*text,x,y*) - Draws text on the canvas (no fill)

```
<script>
	var c=document.getElementById("myCanvas");
	var ctx=c.getContext("2d");
	ctx.font="30px Arial";
	ctx.fillText("Hello World",10,50);
</script>
```

Hello World

Drag and Drop

- Drag and drop is a very common feature. It is when you "grab" an object and drag it to a different location.
- In HTML5, drag and drop is part of the standard, and any element can be draggable.

```
<img draggable="true">
```

Make an Element Draggable
First of all: To make an element draggable, set the draggable attribute to true:

What to Drag - ondragstart and setData()
Then, specify what should happen when the element is dragged.
In the example above, the ondragstart attribute calls a function, drag(event), that specifies what data to be dragged.
The dataTransfer.setData() method sets the data type and the value of the dragged data:

```
function drag(ev) {
    ev.dataTransfer.setData("text", ev.target.id);
}
```

Drag and drop

Where to Drop - ondragover

The ondragover event specifies where the dragged data can be dropped.
By default, data/elements cannot be dropped in other elements. To allow a drop, we must prevent the default handling of the element.
This is done by calling the event.preventDefault() method for the ondragover event:

```
event.preventDefault()
```

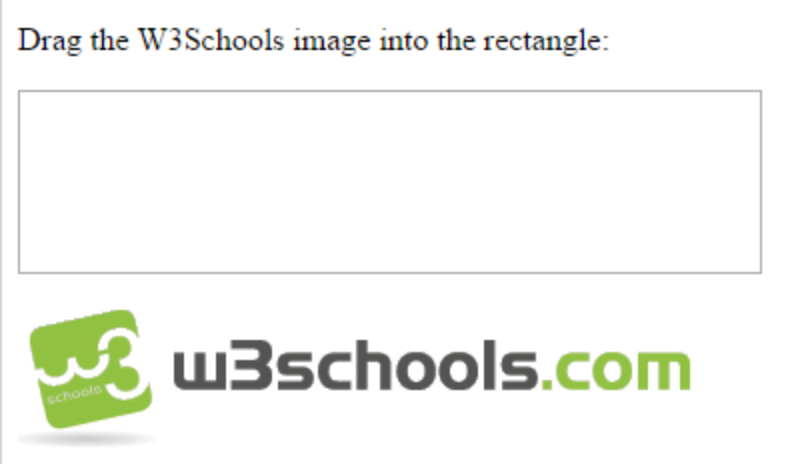
Do the Drop - ondrop

When the dragged data is dropped, a drop event occurs.
In the example above, the ondrop attribute calls a function, drop(event):

```
function drop(ev) {
    ev.preventDefault();
    var data = ev.dataTransfer.getData("text");
    ev.target.appendChild(document.getElementById(data));
}
```

```
<!DOCTYPE HTML>
<html>
<head>
<style>
#div1 {width:350px;height:70px;padding:10px;border:1px solid #aaaaaa;}
</style>
<script>
function allowDrop(ev) {
    ev.preventDefault();
}

function drag(ev) {
    ev.dataTransfer.setData("text", ev.target.id);
}

function drop(ev) {
    ev.preventDefault();
    var data = ev.dataTransfer.getData("text");
    ev.target.appendChild(document.getElementById(data));
}
</script>
</head>
<body>

<p>Drag the W3Schools image into the rectangle:</p>

<div id="div1" ondrop="drop(event)" ondragover="allowDrop(event)"></div>
<br>
<img id="drag1" src="img_logo.gif" draggable="true" ondragstart="drag(event)" width="336" height="69">

</body>
</html>
```

Example:



Drag and drop

Code explained:

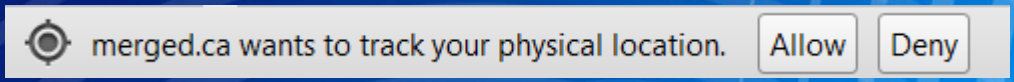
- Call preventDefault() to prevent the browser default handling of the data (default is open as link on drop)

- Get the dragged data with the dataTransfer.getData() method. This method will return any data that was set to the same type in the setData() method

- The dragged data is the id of the dragged element ("drag1")
- Append the dragged element into the drop element

Geolocation

- The HTML5 Geolocation API is used to get the geographical position of a user
- The geolocation API centers around a new property on the global navigator object: navigator.geolocation
- Use the getCurrentPosition() method to get the user's position
- If the getCurrentPosition() method is successful, it returns a coordinates object to the function specified in the first parameter
- Otherwise, the function in the second parameter will be executed



Geolocation

POSITION OBJECT		
Property	Type	Notes
`coords.latitude`	double	decimal degrees
`coords.longitude`	double	decimal degrees
`coords.altitude`	double or `null`	meters above the reference ellipsoid
`coords.accuracy`	double	meters
`coords.altitudeAccuracy`	double or `null`	meters
`coords.heading`	double or `null`	degrees clockwise from true north
`coords.speed`	double or `null`	meters/second
`timestamp`	DOMTimeStamp	like a `Date()` object

Source: http://diveintohtml5.info/geolocation.html

HTML5 Local Storage

- With local storage, web applications can store data locally within the user's browser.
- Before HTML5, application data had to be stored in cookies, included in every server request. Local storage is more secure, and large amounts of data can be stored locally, without affecting website performance.
- Unlike cookies, the storage limit is far larger (at least 5MB) and information is never transferred to the server.
- Local storage is per origin (per domain and protocol). All pages, from one origin, can store and access the same data.

HTML5 Local Storage

HTML Local Storage Objects

- HTML local storage provides two objects for storing data on the client:
  - window.localStorage - stores data with no expiration date
  - window.sessionStorage - stores data for one session (data is lost when the browser tab is closed)
- Before using local storage, check browser support for localStorage and sessionStorage:

```
if(typeof(Storage) !== "undefined") {
    // Code for localStorage/sessionStorage.
} else {
    // Sorry! No Web Storage support..
}
```

HTML5 Local Storage

The local Storage Object

- The localStorage object stores the data with no expiration date. The data will not be deleted when the browser is closed, and will be available the next day, week, or year.

```
// Store
localStorage.setItem("lastname", "Smith");
// Retrieve
document.getElementById("result").innerHTML =
localStorage.getItem("lastname");
```

or

```
// Store
localStorage.lastname = "Smith";
// Retrieve
document.getElementById("result").innerHTML = localStorage.lastname;
```

The syntax for removing the "lastname" localStorage item is as follows:

```
localStorage.removeItem("lastname");
```

HTML5 Local Storage

The session Storage Object

- The sessionStorage object is equal to the localStorage object, **except** that it stores the data for only one session. The data is deleted when the user closes the specific browser tab.
- The following example counts the number of times a user has clicked a button, in the current session:

```
if (sessionStorage.clickcount) {
    sessionStorage.clickcount = Number(sessionStorage.clickcount) + 1;
} else {
    sessionStorage.clickcount = 1;
}
document.getElementById("result").innerHTML = "You have clicked the button
" +
sessionStorage.clickcount + " time(s) in this session.";
```

Guidelines

- Always put closing tags for the ones that need. Do not rely on browser ability to display page with unclosed tag.

- Check whether HTML tags and attributes can support different browsers, versions and platforms.