

Lecture No: 8**Topic: Review: HTML 5****HTML5 template:**

```

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Title of the document</title>
</head>

<body>
Content of the document.....
</body>

</html>

```

New HTML5 Elements

The most interesting new HTML5 elements are:

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

Removed Elements in HTML5

Removed Element	Use Instead
<acronym>	<abbr>
<applet>	<object>
<basefont>	CSS
<big>	CSS
<center>	CSS
<dir>	
	CSS
<frame>	
<frameset>	
<noframes>	
<strike>	CSS, <s>, or
<tt>	CSS

HTML5 Browser Support

- HTML5 is supported in all modern browsers.
- In addition, all browsers, old and new, automatically handle unrecognized elements as inline elements.
- Because of this, you can "teach" older browsers to handle "unknown" HTML elements.

Define Semantic Elements as Block Elements

- HTML5 defines eight new **semantic** elements. All these are **block-level** elements.
- To secure correct behavior in older browsers, you can set the CSS **display** property for these HTML elements to **block**:

```
header, section, footer, aside, nav, main, article, figure {
  display: block;
}
```

New Elements in HTML5

Tag	Description
<article>	Defines an article in a document
<aside>	Defines content aside from the page content
<bdi>	Isolates a part of text that might be formatted in a different direction from other text outside it
<details>	Defines additional details that the user can view or hide
<dialog>	Defines a dialog box or window
<figcaption>	Defines a caption for a <figure> element
<figure>	Defines self-contained content
<footer>	Defines a footer for a document or section
<header>	Defines a header for a document or section
<main>	Defines the main content of a document
<mark>	Defines marked/highlighted text
<meter>	Defines a scalar measurement within a known range (a gauge)
<nav>	Defines navigation links
<progress>	Represents the progress of a task
<rp>	Defines what to show in browsers that do not support ruby annotations
<rt>	Defines an explanation/pronunciation of characters (for East Asian typography)
<ruby>	Defines a ruby annotation (for East Asian typography)
<section>	Defines a section in a document
<summary>	Defines a visible heading for a <details> element
<time>	Defines a date/time
<wbr>	Defines a possible line-break

New Form Elements

Tag	Description
<datalist>	Specifies a list of pre-defined options for input controls
<output>	Defines the result of a calculation

New Input Types

- **New Input Types**
 - color
 - date
 - datetime
 - datetime-local
 - email
 - month
 - Number
 - range

- search
- tel
- time
- url
- week

- **New Input Attributes**

- autocomplete
- autofocus
- form
- formaction
- formenctype
- formmethod
- formnovalidate
- formtarget
- height and width
- list
- min and max
- multiple
- pattern (regexp)
- placeholder
- required
- step

HTML5 - New Attribute Syntax

- HTML5 allows four different syntaxes for attributes.

Type	Example
Empty	<code><input type="text" value="John" disabled></code>
Unquoted	<code><input type="text" value=John></code>
Double-quoted	<code><input type="text" value="John Doe"></code>
Single-quoted	<code><input type="text" value='John Doe'></code>

HTML5 Graphics

Tag	Description
<code><canvas></code>	Draw graphics, on the fly, via scripting (usually JavaScript)
<code><picture></code>	Defines a container for multiple image resources
<code><svg></code>	Draw scalable vector graphics

New Media Elements

Tag	Description
<code><audio></code>	Defines sound content
<code><embed></code>	Defines a container for an external (non-HTML) application
<code><picture></code>	Defines a container for multiple image resources
<code><source></code>	Defines multiple media resources for media elements (<code><video></code> and <code><audio></code>)

<track>	Defines text tracks for media elements (<video> and <audio>)
<video>	Defines video or movie






HTML5 Semantic Elements

- Semantics is the study of the meanings of words and phrases in a language.
- Semantic elements = elements with a meaning.

What are Semantic Elements?

- A semantic element clearly describes its meaning to both the browser and the developer.
- Examples of **non-semantic** elements: <div> and - Tells nothing about its content.
- Examples of **semantic** elements: <form>, <table>, and <article> - Clearly defines its content.

Browser Support

				
Yes	Yes	Yes	Yes	Yes

- HTML5 semantic elements are supported in all modern browsers.

New Semantic Elements in HTML5

- Many web sites contain HTML code like:

```
<div id="nav">
```

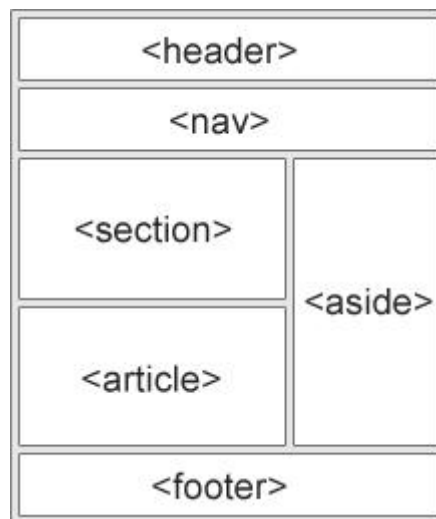
```
<div class="header">
```

```
<div id="footer">
```

to indicate navigation, header, and footer.

- HTML5 offers new semantic elements to define different parts of a web page:

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



HTML5 <section> Element

- The <section> element defines a section in a document.
- According to W3C's HTML5 documentation: "A section is a thematic grouping of content, typically with a heading."
- A home page could normally be split into sections for introduction, content, and contact information.

Example

```
<section>
  <h1>WWF</h1>
  <p>The World Wide Fund for Nature (WWF) is....</p>
</section>
```

HTML5 <article> Element

- The <article> element specifies independent, self-contained content.
- An article should make sense on its own, and it should be possible to read it independently from the rest of the web site.
- Examples of where an <article> element can be used:
 - Forum post
 - Blog post
 - Newspaper article

Example

```
<article>
  <h1>What Does WWF Do?</h1>
  <p>WWF's mission is to stop the degradation of our planet's ....p>
</article>
```

Nesting <article> in <section> or Vice Versa?

- The <article> element specifies independent, self-contained content.
- The <section> element defines section in a document.
- So, on the Internet, you will find HTML pages with <section> elements containing <article> elements, and <article> elements containing <section> elements.
- You will also find pages with <section> elements containing <section> elements, and <article> elements containing <article> elements.
- Example for a newspaper:
- The sport <article> in the sport **section**, may have a technical **section** in each <article>.

HTML5 <header> Element

- The <header> element specifies a header for a document or section.
- The <header> element should be used as a container for introductory content.
- You can have several <header> elements in one document.

Example

```
<article>
  <header>
    <h1>What Does WWF Do?</h1>
    <p>WWF's mission:</p>
```

```
</header>
<p>WWF's mission is to ...>
</article>
```

HTML5 <footer> Element

- The <footer> element specifies a footer for a document or section.
- A <footer> element should contain information about its containing element.
- A footer typically contains the author of the document, copyright information, links to terms of use, contact information, etc.
- You may have several <footer> elements in one document.

Example

```
<footer>
  <p>Posted by: Hege Refsnes</p>
  <p>Contact information: <a href="mailto:someone@example.com">
    someone@example.com</a>.</p>
</footer>
```

HTML5 <nav> Element

- The <nav> element defines a set of navigation links.
- Notice that NOT all links of a document should be inside a <nav> element. The <nav> element is intended only for major block of navigation links.

Example

```
<nav>
  <a href="/html/">HTML</a> |
  <a href="/css/">CSS</a> |
  <a href="/js/">JavaScript</a> |
  <a href="/jquery/">jQuery</a>
</nav>
```

HTML5 <aside> Element

- The <aside> element defines some content aside from the content it is placed in (like a sidebar).
- The <aside> content should be related to the surrounding content.

Example

```
<p>My family and I visited The Epcot center this summer.</p>
<aside>
  <h4>Epcot Center</h4>
  <p>The Epcot Center is a theme park in Disney World, Florida.</p>
</aside>
```

HTML5 <figure> and <figcaption> Elements

- The purpose of a figure caption is to add a visual explanation to an image.
- In HTML5, an image and a caption can be grouped together in a <figure> element.
- The element defines the image, the <figcaption> element defines the caption.

Example

```
<figure>
  
  <figcaption>Fig1. – This is Panda</figcaption>
</figure>
```

Why Semantic Elements?

- With HTML4, developers used their own id/class names to style elements: header, top, bottom, footer, menu, navigation, main, container, content, article, sidebar, topnav, etc.
- This made it impossible for search engines to identify the correct web page content.
- With the new HTML5 elements (<header> <footer> <nav> <section> <article>), this will become easier.
- According to the W3C, a Semantic Web: "Allows data to be shared and reused across applications, enterprises, and communities."

HTML5 Migration

- Migration from HTML4 to HTML5

Typical HTML4	Typical HTML5
<div id="header">	<header>
<div id="menu">	<nav>
<div id="content">	<section>
<div class="article">	<article>
<div id="footer">	<footer>

Change to HTML5 Doctype

- Change the **doctype**:
 - <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
- to the HTML5 doctype:
 - <!DOCTYPE html>

Change to HTML5 Encoding

- Change the **encoding** information:
 - <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
- to HTML5 encoding:
 - <meta charset="utf-8">

Change to HTML5 Semantic Elements

- The existing CSS contains id's and classes for styling the elements:

```
body {
  font-family: Verdana,sans-serif;
  font-size: 0.9em;
}

div#header, div#footer {
  padding: 10px;
  color: white;
  background-color: black;
}

div#content {
  margin: 5px;
  padding: 10px;
  background-color: lightgrey;
}

div.article {
  margin: 5px;
  padding: 10px;
  background-color: white;
}

div#menu ul {
  padding: 0;
}

div#menu ul li {
  display: inline;
  margin: 5px;
}
```

Replace with equal CSS styles for HTML5 semantic elements:

```
body {
  font-family: Verdana,sans-serif;
  font-size: 0.9em;
}
header, footer {
```



```
padding: 10px;
color: white;
background-color: black;
}
section {
margin: 5px;
padding: 10px;
background-color: lightgrey;
}
article {
margin: 5px;
padding: 10px;
background-color: white;
}
nav ul {
padding: 0;
}
nav ul li {
display: inline;
margin: 5px;
}
```

HTML5 Style Guide and Coding Conventions

- HTML Coding Conventions
 - Web developers are often uncertain about the coding style and syntax to use in HTML.
 - Between 2000 and 2010, many web developers converted from HTML to XHTML.
 - With XHTML, developers were forced to write valid and "well-formed" code.
 - HTML5 is a bit more sloppy when it comes to code validation
- Be Smart and Future Proof
 - A consistent use of style makes it easier for others to understand your HTML.
 - In the future, programs like XML readers may want to read your HTML.
 - Using a well-formed-"close to XHTML" syntax can be smart.
 - Always keep your code tidy, clean and well-formed.
- Use Correct Document Type
 - Always declare the document type as the first line in your document:
 - <!DOCTYPE html>
 - If you want consistency with lower case tags, you can use:
 - <!doctype html>
- Use Lower Case Element Names
 - HTML5 allows mixing uppercase and lowercase letters in element names.
 - We recommend using lowercase element names because:
 - Mixing uppercase and lowercase names is bad
 - Developers normally use lowercase names (as in XHTML)
 - Lowercase look cleaner

- Lowercase are easier to write

- **BAD**

```
<SECTION>
```

```
  <p>This is a paragraph.</p>
```

```
</SECTION>
```

- **VERY BAD**

```
<Section>
```

```
  <p>This is a paragraph.</p>
```

```
</SECTION>
```

- **GOOD**

```
<section>
```

```
  <p>This is a paragraph.</p>
```

```
</section>
```

Close All HTML Elements

- **BAD**

```
<section>
```

```
  <p>This is a paragraph.
```

```
  <p>This is a paragraph.
```

```
</section>
```

- **GOOD**

```
<section>
```

```
  <p>This is a paragraph.</p>
```

```
  <p>This is a paragraph.</p>
```

```
</section>
```

Close Empty HTML Elements

- In HTML5, it is optional to close empty elements.

- **Allowed:**

```
<meta charset="utf-8">
```

- **Also Allowed:**

```
<meta charset="utf-8" />
```

- However, the closing slash (/) is REQUIRED in XHTML and XML.
- If you expect XML software to access your page, it is a good idea to keep the closing slash.

- Use Lower Case Attribute Names

- HTML5 allows mixing uppercase and lowercase letters in attribute names.
- We recommend using lowercase attribute names because:
 - Mixing uppercase and lowercase names is bad
 - Developers normally use lowercase names (as in XHTML)
 - Lowercase look cleaner
 - Lowercase are easier to write

- Bad:

```
<div CLASS="menu">
```

- Good:


```
<div class="menu">
```
- Quote Attribute Values
 - HTML5 allows attribute values without quotes.
 - We recommend quoting attribute values because:
 - Developers normally quote attribute values (as in XHTML)
 - Quoted values are easier to read
 - You MUST use quotes if the value contains spaces
- Very bad:
 - This will not work, because the value contains spaces:


```
<table class=table striped>
```
- Bad:


```
<table class=striped>
```
- Good:


```
<table class="striped">
```
- Image Attributes
 - Always add the alt attribute to images. This attribute is important when the image for some reason cannot be displayed. Also, always define image width and height. It reduces flickering because the browser can reserve space for the image before loading.
- Bad:


```

```
- Good:


```

```
- Spaces and Equal Signs
 - HTML5 allows spaces around equal signs. But space-less is easier to read and groups entities better together.
- Bad:


```
<link rel = "stylesheet" href = "styles.css">
```
- Good:


```
<link rel="stylesheet" href="styles.css">
```
- Avoid Long Code Lines
 - When using an HTML editor, it is inconvenient to scroll right and left to read the HTML code.
 - Try to avoid code lines longer than 80 characters.
- Blank Lines and Indentation
 - Do not add blank lines without a reason.
 - For readability, add blank lines to separate large or logical code blocks.
 - For readability, add two spaces of indentation. Do not use the tab key.
 - Do not use unnecessary blank lines and indentation. It is not necessary to indent every element:
- **Unnecessary:**

```
<body>
```

```
<h1>Famous Cities</h1>
```

```
<h2>Tokyo</h2>
```

```
<p>
```

Tokyo is the capital of Japan, the center of the Greater Tokyo Area,
and the most populous metropolitan area in the world.

It is the seat of the Japanese government and the Imperial Palace,
and the home of the Japanese Imperial Family.

```
</p>
```

```
</body>
```

- **Better:**

```
<body>
```

```
<h1>Famous Cities</h1>
```

```
<h2>Tokyo</h2>
```

```
<p>Tokyo is the capital of Japan, the center of the Greater Tokyo Area,  
and the most populous metropolitan area in the world.
```

```
It is the seat of the Japanese government and the Imperial Palace,  
and the home of the Japanese Imperial Family.</p>
```

```
</body>
```

References:

- www.w3schools.com