Introduction to HTML5

Tag Structure

DOCTYPE

The doctype for HTML 4.01

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

The doctype for HTML5

```
<!DOCTYPE html>
```

Tag Structure

<link> Tag

```
<link rel="stylesheet" type="text/css" href="file.css">
```

In HTML5

```
<link rel="stylesheet" href="file.css">
```

Obsolete Elements In HTML5

Frame , frameset , noframes , font , big , center and few more.
Attributes like bgcolor, cellspacing, cellpadding, and valign.

Use of <a> Tag in HTML5

```
<a href="/about">
    <h2>About me</h2>
    <p>Find out what makes me tick.</p>
</a>
```

Tag Structure

Organizing Code Using Blocking Elements

- SECTION
- ARTICLE
- HEADER
- FOOTER
- ASIDE
- FIGURE
- NAV

TechJini™

Forms in HTML5

- **Form elements can be anywhere and can be associated to form by giving form's ID to form attribute of that element.**

```
<form id=foo>
<input type=”text”>
...
</form>
<textarea form=foo></textarea>
```

Forms in HTML5

New INPUT types

Email input type

```
<input type=email>
```

URL input type

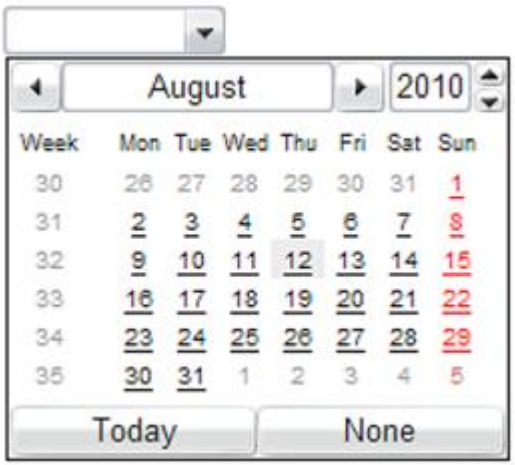
```
<input type=url>
```

Date input type

```
<input type=date>
```

Time input type

```
<input type=time>
```



datetime input type

```
<input type=datetime>
```

Month input type

```
<input type=month>
```

Forms in HTML5

More INPUT types

Week input type

<input type=week>

Number input type

<input type=number min=0 max=20 step=2 >

In Opera

range input type

<input type=range >

In Chrome

Search input type

<input type=search>

tel input type

<input type=tel>

Forms in HTML5

Last INPUT type

color input type

```
<input type=color >
```

In Blackberry web browser

Forms in HTML5

New attributes

Autofocus attribute

```
<input id="status" name="status" type="text" autofocus>
```

Placeholder attribute

```
<label for="search">My Search</label>

<input id="search" name="search" type="text"
placeholder="Search Here">
```

My Search Search Here

Forms in HTML5

More new attributes

Required attribute

```
<input id="pass" name="pass" type="password" required>
```

Autocomplete attribute

```
<input type="text" name="onetimetoken" autocomplete="off">
```

Offline Web Applications

- Allow you to keep using web apps and sites without a network connection
- Browsers cache data in Application cache
- Once resources are cached you can access them very quickly (without network request)
- HTML5 also allows online and offline detection
- We can use offline mechanism that allows us to easily prefetch site resources

TechJini™

Offline Web Applications

The cache manifest

```
<!DOCTYPE HTML>
<html manifest="/cache.manifest">
<body>
...
</body>
</html>
```

Contents of cache.manifest

```
CACHE MANIFEST
# Files to cache
index.html
cache.html
html5.css
image1.jpg
html5.js

# Use from Network if available
NETWORK:
network.html

# Fallback content
FALLBACK:
fallback.js fallbackserver.js
/fallback.html
```

Canvas

- The canvas element provides scripts with a resolution-dependent bitmap canvas, which can be used for rendering graphs, game graphics, or other visual images on the fly.
- Canvas gives an easy and powerful way to draw graphics using Javascript.
- For each canvas element you can use a "context" (think about a page in a drawing pad), into which you can issue JavaScript commands to draw anything you want.

Canvas

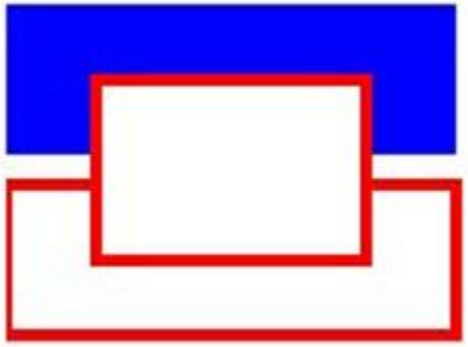
Basics of Canvas

```
<canvas id="myCanvas" width="300" height="150">
Fallback content, in case the browser does not support Canvas.
</canvas>

<script>
// Get a reference to the element.
var elem = document.getElementById('myCanvas');

// Always check for properties and methods, to make
// sure your code doesn't break
// in other browsers.
if (elem && elem.getContext) {
    // Get the 2d context.
    // Remember: you can only initialize one context per element.
    var context = elem.getContext('2d');
    if (context) {
     // You are done! Now you can draw your first rectangle.
     // You only need to provide the (x,y) coordinates, followed by
     // the width and
     // height dimensions.
     context.fillRect(0, 0, 150, 100);
    }
}
</script>
```

Canvas

Basic lines and strokes

```
context.fillStyle   = '#00f'; // blue
context.strokeStyle = '#f00'; // red
context.lineWidth   = 4;

// Draw some rectangles.
context.fillRect  (0,   0, 150, 50);
context.strokeRect(0,  60, 150, 50);
context.clearRect (30, 25,  90, 60);
context.strokeRect(30, 25,  90, 60);
```



Canvas

Text in Canvas

```
context.fillStyle    = '#00f';
context.font         = 'italic 30px sans-serif';
context.textBaseline = 'top';
context.fillText('Hello world!', 0, 0);
context.font         = 'bold 30px sans-serif';
context.strokeText('Hello world!', 0, 50);
```

Hello world!

Hello world!

Videos

- HTML5 provides a standardised way to play video directly in the browser, with no plugins required.
- No <object> , <embed> elements required.
- <video> elements can be styled with CSS
- HTML5 can be tweaked and redisplayed onto <canvas> with Javascript.

Videos

- Usage of video element

```
<video src=turkish.ogv></video>
```

- Fallback markup between the tags, for older Web browsers that do not support native video

```
<h1>Video and legacy browser fallback</h1>
<video src=leverage-a-synergy.ogv>
     Download the <a href=leverage-a-synergy.ogv>How to
     leverage a synergy video</a>
</video>
```

Videos

Attributes supported

- **autoplay**

```
<video src=leverage-a-synergy.ogv autoplay>
</video>
```

It tells the browser to play the video or audio automatically.

- **controls**

```
<video src=leverage-a-synergy.ogv controls>
</video>
```

Provides play/ pause toggle, a seek bar, and volume control.

TechJini™

Videos

Attributes supported

- **poster**

```
<video src=leverage-a-synergy.ogv poster >
</video>
```

The poster attribute points to an image that the browser will use while the video is downloading, or until the user tells the video to play.

- **height , width**

These attributes tell the browser the size in pixels of the video.

TechJini™

Videos

Attributes supported

- **loop**

```
<video src=leverage-a-synergy.ogv loop >
</video>
```

It loops the media playback.

- **preload (preload=auto)**

Indicates browser that it should begin downloading the entire file.

Videos

Attributes supported

- **preload = none**

```
<video src=leverage-a-synergy.ogv preload=none >
</video>
```

This state suggests to the browser that it shouldn’t preload the resource until the user activates the controls.