HTML5: New Standards for Web Interactivity

By James Sugrue

CONTENTS INCLUDE:
- About HTML5
- Changes to Existing Components
- New Elements in HTML5
- Attribute Changes
- HTML Event Handling
- Hot Tips and more...

ABOUT HTML5

HTML5 is a standard for structuring and presenting content on the Web. It incorporates features such as geolocation, video playback and drag-and-drop. HTML5 allows developers to create rich internet applications without the need for third party APIs and browser plug-ins.

HTML5 is still under specification, and is currently in the Working Draft stage in the W3C, but many aspects of HTML5 are now stable and can be implemented in browsers.

This DZone Refcard highlights the main features in HTML5 and illustrates the JavaScript APIs available to work with those features.

CHANGES TO EXISTING COMPONENTS

Simplified Syntax
There are a number of simplifications to the syntax of HTML introduced in the HTML5 specification.

Document Type
The `<doctype>` for an HTML document has changed from its verbose DTD reference to a much simpler format, simply stating the document is an HTML document type:

```html
<!doctype html>
```

This change allows HTML5 to be fully backward compatible with previous versions of HTML.

Character Encoding
The `<meta>` tag for a document allows you to set the character encoding using the simple `charset` attribute, replacing declarations such as:

```html
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
```

with

```html
<meta charset=UTF-8">
```

Character Encoding
The `<meta>` tag for a document allows you to set the character encoding using the simple `charset` attribute, replacing declarations such as:

```html
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
```

with

```html
<meta charset=UTF-8">
```

Script and Link Elements
The `<script>` element has been stripped down, removing the need for the type attribute. The reason for this is that scripts are typically written in JavaScript. The `<link>` element has lost its type attribute due to the prevalence of CSS.

NEW ELEMENTS IN HTML5

The following table presents an overview of the new elements that have been added to HTML5.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;article&gt;</code></td>
<td>An independent piece of content for a document e.g. blog entry, forum entry</td>
</tr>
<tr>
<td><code>&lt;aside&gt;</code></td>
<td>A piece of content that is somehow related to the rest of the page</td>
</tr>
<tr>
<td><code>&lt;audio&gt;</code></td>
<td>Audio media content</td>
</tr>
<tr>
<td><code>&lt;canvas&gt;</code></td>
<td>A component for rendering dynamic bitmap graphics on the fly, e.g. games</td>
</tr>
<tr>
<td><code>&lt;command&gt;</code></td>
<td>A command that the user can invoke: a button, radio button or checkbox</td>
</tr>
<tr>
<td><code>&lt;datalist&gt;</code></td>
<td>Together with the new list attribute for the <code>&lt;input&gt;</code> element can be used to make combo boxes</td>
</tr>
<tr>
<td><code>&lt;details&gt;</code></td>
<td>Additional information or controls that the user can obtain on demand, to provide details on the document, or parts of it</td>
</tr>
<tr>
<td><code>&lt;embed&gt;</code></td>
<td>Used for plug-in content</td>
</tr>
<tr>
<td><code>&lt;figure&gt;</code></td>
<td>A piece of self-contained flow content referenced as a single unit from the main flow of the document</td>
</tr>
<tr>
<td><code>&lt;figcaption&gt;</code></td>
<td>Caption for a <code>&lt;figure&gt;</code></td>
</tr>
<tr>
<td><code>&lt;footer&gt;</code></td>
<td>Footer for a section; may contain information about author, copyright information, etc.</td>
</tr>
<tr>
<td><code>&lt;header&gt;</code></td>
<td>A group of introductory or navigation aids</td>
</tr>
<tr>
<td><code>&lt;hgroup&gt;</code></td>
<td>Header of a section</td>
</tr>
<tr>
<td><code>&lt;keygen&gt;</code></td>
<td>A key pair generation control for user authentication in forms</td>
</tr>
<tr>
<td><code>&lt;mark&gt;</code></td>
<td>A run of text in one document marker or highlighted for reference purposes</td>
</tr>
</tbody>
</table>

Other Encoding Options
You can also set the encoding of the document by setting the Byte Order Mark at the start of the file, or by setting the HTTP Content-Type header at transport level.

Don’t Miss An Issue!
Get over 90 DZone Refcardz FREE from Refcardz.com!

Visit Refcardz.com to get them all Free!
<abbr> A measurement, such as disk usage, when the minimum and maximum values are known.
<nav> A section of the document intended for navigation.
<output> Output such as a calculation done through scripting.
<progress> Represents progress of a task such as downloading or performing other expensive operations.
<rt> Enables Ruby annotation markup. &lt;rt&gt; gives an explanation of the Ruby annotation &lt;rbr&gt; is what the browser should show if it does not support the &lt;rruby&gt; element.
<section> A generic document or application section.
<source> Used to specify multiple media resources on elements such as &lt;audio&gt; and &lt;video&gt;.
<time> Date and time definition.
<video> Video media content.

**ATTRIBUTE CHANGES**

The following section presents an overview of the attributes that have been added or changed in existing HTML elements.

**Global Attributes**
The following global attributes have been added for use across a number of elements.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>contenteditable</td>
<td>Indicates that the element is an editable area</td>
</tr>
<tr>
<td>contextmenu</td>
<td>Points to a context menu provided by the author</td>
</tr>
<tr>
<td>data-*</td>
<td>All author defined attributes need to be prefixed by data-, preventing clashes with future versions of HTML</td>
</tr>
<tr>
<td>draggable</td>
<td>Used with the drag &amp; drop JavaScript API</td>
</tr>
<tr>
<td>hidden</td>
<td>Indicates that an element is not relevant</td>
</tr>
<tr>
<td>role aria-*</td>
<td>Instructs assistive technology</td>
</tr>
<tr>
<td>spellcheck</td>
<td>Enables spell check indicators if content can be checked</td>
</tr>
</tbody>
</table>

**Input Element**
The input element’s type attribute now has these new attributes: color, date, datetime, datetime-local, email, month, number, range, search and tel.

**Presentational Elements**

HTML5 has removed the use of presentational attributes such as align, background (for body), bgcolor and border, as they are better handled in CSS.

**REMOVED ELEMENTS**

The following elements have been removed from HTML5 because they are more effectively represented using CSS: basefont, big, center, font, s, strike, tt and u.

Other elements have been removed because they have a negative effect on usability and accessibility. These include: frame, frameset andnoframes.

This last set of elements has been removed due to their lack of frequent use. They also caused confusion at times: acronym (use abbr for abbreviations), applet (object replaces its use), isIndex and dir (use ul instead).

** Canvas **

&lt;canvas&gt; is probably the most dramatic element to be added to HTML5. It allows graphics on the client’s browser to be dynamically updated.

**HTML Representation**
The &lt;canvas&gt; element has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>height</td>
<td>The height of the canvas, specified in pixels. Default is 150</td>
</tr>
<tr>
<td>id</td>
<td>An identifier for the canvas so that it is accessible via Javascript</td>
</tr>
<tr>
<td>width</td>
<td>The width of the canvas, specified in pixels. Default is 300</td>
</tr>
</tbody>
</table>

**JavaScript API**
All dynamic behavior for the canvas is specified in JavaScript. Once you have access to the canvas element, you must create a context in which the canvas can be drawn on.

The following code snippet shows how to access the Canvas and retrieve its graphical context.

```javascript
var canvas = document.getElementById("documentCanvas");
var context = canvas.getContext("2d");
```

Currently the only valid context for the Canvas element is 2D. There will probably be a 3D context in the future. The Context object has the a number of different methods. These include transformations, compositing, state, coloring and shape methods.

**Transformations**
When creating shapes and paths, transformations are applied to coordinates. Transformations must be performed in reverse order.

**Transformation Methods**

- `context.rotate(angle)` Changes the transformation matrix to apply a clockwise rotation expressed in radians.
- `context.scale(x,y)` Applies a scaling transformation, where x represents scale factor in the horizontal direction and y represents the scale factor in the vertical direction, specified in multiples.
- `context.translate(x,y)` Applies a translation transformation to the context where x is the distance along the horizontal axis and y is the distance along the vertical axis, specified in coordinate space units.

**Context State**
A context has a stack of drawing states including:

- The current transformation matrix
- The current clipping region
- Values for a number of attributes

The following methods can be used to manage state:

**Context State Methods**

- `context.restore()` Pops the top state on the stack and acts as an undo action and restores the context to that state
- `context.save()` Pushes the current state onto the stack
Compositing

All drawing operations are affected by the global compositing values.

Compositing Methods

context.globalAlpha [=value]
Returns or sets the current alpha value applied to rendering operations. Values outside the range 0.0 (fully transparent) to 1.0 (no transparency) are ignored.

context.globalCompositeOperation [=value]
Returns or sets the current composition operation from the list below:
• copy
  Display source image instead of destination image
• destination-atop
  Display destination image wherever both are opaque. Display source image where source is opaque but destination is transparent
• destination-in
  Display destination image wherever both destination and source images are opaque
• destination-out
  Display destination image wherever destination image is opaque and source image is transparent
• destination-over
  Display destination image wherever destination image is opaque. Source image elsewhere
• source-atop
  Display the sum of source and destination images
• source-in
  Display source image wherever both source image and destination image are opaque
• source-out
  Display source image wherever source image is opaque and destination image is transparent
• source-over (default)
  Display source image wherever source image is opaque and destination image is opaque but destination is transparent
• vendorName-operationName
  Vendor specific operations follow this format
• xor
  Exclusive OR of source and destination image

Colors and Styles

Style Methods

context.fillStyle [=value]
Returns or sets the current style used for filling shapes. Can be a string containing a CSS color, or a CanvasGradient/CanvasPattern

context.strokeStyle [=value]
Returns or sets the current style used for the stroke of shapes. Can be a string containing a CSS color, or a CanvasGradient/CanvasPattern

To create the appropriate CanvasGradient object use either either createLinearGradient() or createRadialGradient() from the Context object. The resulting CanvasGradient object has the following method available:

CanvasGradient Methods

gradient.addColorStop(offset, color)
Adds a new stop to the gradient at a point between 0.0 and 1.0, each representing each end of the gradient. The color parameter must be a CSS color.

To create a CanvasPattern object use the following method:

CanvasPattern Method

context.createPattern(image, repetition)
Creates a new CanvasPattern object using the given image. The image is repeated in one of these specified directions:
• repeat: both directions
• repeat-x: horizontal only
• repeat-y: vertical only
• no-repeat: neither
The image can be either an img, video or canvas.

Line Styles

The following line styles can be applied from the context:

Line Style Methods

context.lineDash=[value]
The ending that will be placed at the end of the line. Either butt, round or square. Initially set to butt

context.lineCap=[value]
The corners that will be placed where two lines meet. Either bevel, round or miter. Initially set to miter

context.lineWidth=[value]
Width of the line, in coordinate space units. Initially set to 1.0

context.miterLimit=[value]
Maximum allowed ratio of the miter length to half the line width. Miter length is the distance from the point where the join occurs to the intersection of the line edges on the outside of the join. Initially set to 10.0.

Shadows

There are four global shadow attributes that affect all drawing operations:

Shadow Methods

context.shadowBlur [=value]
The size of the blurring effect, initially set to 0.

context.shadowColor [=value]
The color of the shadow, initially transparent black. Specified as a CSS color.

context.shadowOffsetX [=value]
context.shadowOffsetY [=value]
The distance that a shadow will be offset in horizontal and vertical distance

Shapes

Three methods are available to create simple shapes, or rectangles:

Simple Shape Methods

context.clearRect(x,y,w,h)
Clears the pixels in the specified rectangle that also intersect the current clipping region to fully transparent black, erasing the previous image

context.fillRect(x,y,w,h)
Paints the specified rectangle using the defined fillStyle

context.strokeRect(x,y,w,h)
Draws the outline of the rectangle path using strokeStyle, lineWidth, lineJoin and miterLimit attributes

The context always has a current path, which can have zero or more sub-paths. These are used to create complex shapes.

Complex Shape Methods

context.arc(x,y,radius)
context.arcTo(x1,y1,x2,y2, radius)
context.bezierCurveTo(cp1x,cp1y,cp2x,cp2y,xx,yy)
context.clip()
context.closePath()
context.beginPath()
context.moveTo(x,y)
context.lineTo(x,y)
context.quadraticCurveTo(cp1x,cp1y,xx,yy)
context.rect(x,y,w,h)
context.stroke(path)
context.strokeRect(x,y,w,h)
context.fillPath()
context.setFillstyle()
context.lineTo(x,y)
context.moveTo(x,y)
context.strokeTo(x,y)
context.strokeRect(x,y,w,h)
context.setFill(style)
context.stroke(x,y)
context.move(x,y)
context.lineTo(x,y)
context.strokeTo(x,y)
context.closePath()
context.beginPath()
context.moveTo(x,y)
context.lineTo(x,y)
context.quadraticCurveTo(cp1x,cp1y,cp2x,cp2y,xx,yy)
context.rect(x,y,w,h)
context.stroke(x,y)
context.fill(x,y)
context.strokeTo(x,y)
context.closePath()
context.beginPath()
context.moveTo(x,y)
context.lineTo(x,y)
context.quadraticCurveTo(cp1x,cp1y,cp2x,cp2y,xx,yy)
context.rect(x,y,w,h)
context.stroke(x,y)
context.fill(x,y)
context.strokeTo(x,y)
context.closePath()
context.beginPath()
context.moveTo(x,y)
context.lineTo(x,y)
context.quadraticCurveTo(cp1x,cp1y,cp2x,cp2y,xx,yy)
context.rect(x,y,w,h)
context.stroke(x,y)
context.fill(x,y)
context.strokeTo(x,y)
The video element has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>autoplay</td>
<td>Specifies that the video will start playing as soon as the content is loaded. Value of the attribute should be &quot;autoplay&quot;</td>
</tr>
<tr>
<td>controls</td>
<td>Specifies play button and volume widget controls will be displayed. Value of the attribute should be &quot;controls&quot;</td>
</tr>
<tr>
<td>height</td>
<td>The height of the video player, specified in pixels</td>
</tr>
<tr>
<td>loop</td>
<td>Specifies that the video will play in a continuous loop. Value of the attribute should be &quot;loop&quot;</td>
</tr>
<tr>
<td>preload</td>
<td>Specifies that the video should load when the page loads. Ignored if autoplay attribute is used. Value of the attribute should be &quot;preload&quot;</td>
</tr>
<tr>
<td>src</td>
<td>The URL of the video to play</td>
</tr>
<tr>
<td>width</td>
<td>The width of the video player, specified in pixels</td>
</tr>
</tbody>
</table>

The following example shows how to specify a video for a page with two different formats. This is for wider browser support. Note that you can specify a message to display within the video tag for browsers that do not support the tag.

```html
<video width="800" height="600" controls="controls"
  autoplay="autoplay">
  <source src="http://mysite.com/movie.ogv" type="video/ogg" />
  <source src="http://mysite.com/movie.mp4" type="video/mp4" />
</video>
```

Audio

The `<audio>` element allows the embedding of audio media directly into your HTML page, rather than using third party plug-ins.

**Audio Formats**
The three supported audio formats are MP3, Wav and Ogg Vorbis.

The audio element has the following attributes:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>autoplay</td>
<td>Specifies that the audio will start playing as soon as the content is loaded. Value of the attribute should be &quot;autoplay&quot;</td>
</tr>
<tr>
<td>controls</td>
<td>Specifies that the play button and volume widget controls will be displayed. Value of the attribute should be &quot;controls&quot;</td>
</tr>
<tr>
<td>loop</td>
<td>Specifies that the audio will play in a continuous loop. Value of the attribute should be &quot;loop&quot;</td>
</tr>
<tr>
<td>preload</td>
<td>Specifies that the audio should load when the page loads. Ignored if autoplay attribute is used. Value of the attribute should be &quot;preload&quot;</td>
</tr>
<tr>
<td>src</td>
<td>The URL of the audio file to play</td>
</tr>
</tbody>
</table>

The following example shows how to specify an audio file for an HTML page.

```html
<audio controls="controls" autoplay="autoplay">
  <source src="http://mysite.com/audio.mp3" type="audio/mpeg" />
  <source src="http://mysite.com/audio.ogg" type="audio/ogg" />
</audio>
```

Local Offline Storage

As opposed to using cookies, client data can be stored in two ways—either using longer term localStorage or with single session based sessionStorage. Cookies have traditionally passed data on every server request, but in HTML5 data is passed, only when requested.

To utilize data storage and retrieval you will need to use JavaScript calls.
### HTML5 Event Handling

HTML5 defines a number of new event handling attributes. They are listed here.

#### Media Events
The following events are applicable to all media elements such as audio, video, embed, and img.

<table>
<thead>
<tr>
<th>Event Attribute</th>
<th>Trigger</th>
</tr>
</thead>
<tbody>
<tr>
<td>oncanplay</td>
<td>Media can start to play, but might has to stop for buffering</td>
</tr>
<tr>
<td>oncanplaythrough</td>
<td>Media doesn’t need to wait for buffering</td>
</tr>
<tr>
<td>ondurationchange</td>
<td>Length of media has changed</td>
</tr>
<tr>
<td>onemptied</td>
<td>Media resource becomes “empty” due to a loss of network connection or other errors</td>
</tr>
<tr>
<td>onended</td>
<td>Media has reached the end</td>
</tr>
<tr>
<td>onerror</td>
<td>Error occurs during loading of media</td>
</tr>
<tr>
<td>onloadeddata</td>
<td>Media has loaded</td>
</tr>
<tr>
<td>onloadedmetadata</td>
<td>Meta data of media, such as duration, has loaded</td>
</tr>
<tr>
<td>onloadstart</td>
<td>Browser starts to load data</td>
</tr>
<tr>
<td>onpause</td>
<td>Media has paused</td>
</tr>
<tr>
<td>onplay</td>
<td>Media is about to play</td>
</tr>
<tr>
<td>onplaying</td>
<td>Media is has started playing</td>
</tr>
<tr>
<td>onprogress</td>
<td>Browser is retrieving media data</td>
</tr>
<tr>
<td>onratechange</td>
<td>Playing rate has changed</td>
</tr>
<tr>
<td>onreaddatechanged</td>
<td>Ready-state changed</td>
</tr>
<tr>
<td>onseekend</td>
<td>When seeking has ended (media seeking attribute is no longer true)</td>
</tr>
<tr>
<td>onseeking</td>
<td>When seeking has begun (seeking attribute is true)</td>
</tr>
<tr>
<td>onstalled</td>
<td>Error retrieving media data</td>
</tr>
<tr>
<td>onsuspend</td>
<td>Retrieval of data stopped before completion</td>
</tr>
<tr>
<td>onupdate</td>
<td>Media changes playing position</td>
</tr>
<tr>
<td>onvoleumchange</td>
<td>Media volume changed</td>
</tr>
<tr>
<td>onwaiting</td>
<td>Media stopped playing but should resume</td>
</tr>
</tbody>
</table>

#### Window Events
The following new window events are applicable to the `<body>` element:

<table>
<thead>
<tr>
<th>Event Attribute</th>
<th>Trigger</th>
</tr>
</thead>
<tbody>
<tr>
<td>onafterprint</td>
<td>Document has printed</td>
</tr>
<tr>
<td>onbeforeprint</td>
<td>Document is about to be printed</td>
</tr>
</tbody>
</table>

#### Form Events
The following new event attributes are applicable to form elements, but can be used across other elements:

<table>
<thead>
<tr>
<th>Event Attribute</th>
<th>Trigger</th>
</tr>
</thead>
<tbody>
<tr>
<td>oncontextmenu</td>
<td>Context menu is invoked</td>
</tr>
<tr>
<td>onformchange</td>
<td>Form changes</td>
</tr>
<tr>
<td>onforminput</td>
<td>User inputs data into form</td>
</tr>
<tr>
<td>oninput</td>
<td>Element gets user input</td>
</tr>
<tr>
<td>oninvalid</td>
<td>Element is invalid</td>
</tr>
</tbody>
</table>

#### Session Storage
Session storage should be used only when you require data for a single session. The `sessionStorage` object is used for this. Any attribute can be written to within this object:

```javascript
sessionStorage.username = "James"
```

Retrieval of the same attribute:

```javascript
document.write("Welcome back" + sessionStorage.username);
```

#### Local Storage
Local storage should be used when you have more persistent data with no time limit. Data can be accessed using the same approach shown above using the `localStorage` object.

#### Data Access
A site can only access data that it has previously stored. This prevents cases where different websites can access other site’s data.

#### Mouse Events
Mouse events can happen with any element on the page. The main changes include support for drag and drop.

<table>
<thead>
<tr>
<th>Event Attribute</th>
<th>Trigger</th>
</tr>
</thead>
<tbody>
<tr>
<td>ondrag</td>
<td>Element is dragged</td>
</tr>
<tr>
<td>ondragend</td>
<td>Drag operation is completed</td>
</tr>
<tr>
<td>ondragenter</td>
<td>Element is dragged to a valid drop target</td>
</tr>
<tr>
<td>ondragleave</td>
<td>Element leaves drop target</td>
</tr>
<tr>
<td>ondragover</td>
<td>Element dragged over drop target</td>
</tr>
<tr>
<td>ondragstart</td>
<td>Drag operation has begun</td>
</tr>
<tr>
<td>ondrop</td>
<td>Element is dropped onto drop target</td>
</tr>
<tr>
<td>onmousemove</td>
<td>Mouse wheel is rotated</td>
</tr>
<tr>
<td>onscroll</td>
<td>Element’s scrollbar is being scrolled</td>
</tr>
</tbody>
</table>

#### Form Events
The following new event attributes are applicable to form elements, but can be used across other elements:

<table>
<thead>
<tr>
<th>Event Attribute</th>
<th>Trigger</th>
</tr>
</thead>
<tbody>
<tr>
<td>oncontextmenu</td>
<td>Context menu is invoked</td>
</tr>
<tr>
<td>onformchange</td>
<td>Form changes</td>
</tr>
<tr>
<td>onforminput</td>
<td>User inputs data into form</td>
</tr>
<tr>
<td>oninput</td>
<td>Element gets user input</td>
</tr>
<tr>
<td>oninvalid</td>
<td>Element is invalid</td>
</tr>
</tbody>
</table>

#### GEOLOCATION

The Geolocation API allows you to share your location with trusted websites. This is achieved by retrieving the longitude (long) and latitude through JavaScript.

The global `navigator` object has access to the Geolocation object, which has the following API:

#### Geolocation Interface

```javascript
getGeolocation(positionOptions, successCallback, errorCallbact, in optional PositionOptions options)
```

Asynchronously attempts to obtain the current location of the device. If successful, the successCallback is invoked with a new Position object.

```javascript
long watchGeolocation(positionOptions, successCallback, errorCallbact, in optional PositionOptions options)
```

Immediately returns a long representing the id of a watch operation, and asynchronously handles the watch operation which tracks the position of the device.

```javascript
clearWatch(id)
```

Stops the watch with the supplied identifier. If no such watch exists it simply returns.

The `PositionOptions` object has three attributes:
ABOUT THE AUTHOR

James Sugue has been editor at both Javalobby and Eclipse Zone for over two years, and loves every minute of it. By day, James is a software architect at Pilz Ireland, developing killer desktop software using Java and Eclipse all the way. While working on desktop technologies such as Eclipse RCP and Swing, James also likes meddling with up-and-coming technologies such as Eclipse 4. His current obsession is developing for the iPhone and iPad, having convinced himself that it’s a turning point for the software industry.

RECOMMENDED BOOK

If you don’t know about the new features available in HTML5, now’s the time to find out. This book provides practical information about how and why the latest version of this markup language will significantly change the way you develop for the Web. HTML5: Up & Running carefully guides you through the important changes in this version with lots of hands-on examples, including markup, graphics, and screenshots. You’ll learn how to use HTML5 markup to add video, offline capabilities, and more—and you’ll be able to put that functionality to work right away.

BROWSER SUPPORT FOR HTML5 FEATURES

The table below illustrates when each major browser gained support for key HTML5 features. The numbers indicate the browser’s version when support for each HTML5 feature began.

<table>
<thead>
<tr>
<th>Feature</th>
<th>IE</th>
<th>Firefox</th>
<th>Safari</th>
<th>Opera</th>
<th>Chrome</th>
</tr>
</thead>
<tbody>
<tr>
<td>contenteditable</td>
<td>6.0+</td>
<td>3.5+</td>
<td>3.2+</td>
<td>10.1+</td>
<td>5.0+</td>
</tr>
<tr>
<td>canvas</td>
<td>9.0+</td>
<td>3.0+</td>
<td>3.2+</td>
<td>10.1+</td>
<td>5.0+</td>
</tr>
<tr>
<td>offline storage</td>
<td>8.0+</td>
<td>3.0+</td>
<td>4.0+</td>
<td>10.5+</td>
<td>5.0+</td>
</tr>
<tr>
<td>audio</td>
<td>9.0+</td>
<td>3.5+</td>
<td>4.0+</td>
<td>10.1+</td>
<td>5.0+</td>
</tr>
<tr>
<td>video</td>
<td>9.0+</td>
<td>3.5+</td>
<td>3.2+</td>
<td>10.5+</td>
<td>5.0+</td>
</tr>
<tr>
<td>Canvas Text API</td>
<td>9.0</td>
<td>3.5+</td>
<td>4.0+</td>
<td>10.5+</td>
<td>5.0+</td>
</tr>
<tr>
<td>Drag and Drop</td>
<td>6.0</td>
<td>3.5+</td>
<td>4.0+</td>
<td>-</td>
<td>5.0+</td>
</tr>
<tr>
<td>Geolocation</td>
<td>9.0+</td>
<td>3.5+</td>
<td>5.0+</td>
<td>10.6+</td>
<td>5.0+</td>
</tr>
</tbody>
</table>

Compatibility

http://caniuse.com and https://developers.google.com/web/ are excellent resources to view the compatibility of HTML5 features across a range of browsers.

ABOUT CONTINUOUS INTEGRATION

Continuous Integration can be explained via patterns (i.e., a solution to a problem that appears to be beneficial, but, in the end, they tend to produce more hassles than they solve). For example, a common pattern is to do things like “Build Software at Every Change.”

Continuous Integration refers to the “build and test” cycle, this is what PC Magazine says.

Tutorials, cheat sheets, blogs, feature articles, source code and more... DZone offers something for everyone, including news, tutorials, cheat sheets, blogs, feature articles, source code and more. DZone communities deliver over 6 million pages each month to more than 3.3 million software developers, architects and decision makers. DZone offers something for everyone, including news, tutorials, cheat sheets, blogs, feature articles, source code and more.

“DZone is a developer’s dream,” says PC Magazine.

DZone communities deliver over 6 million pages each month to more than 3.3 million software developers, architects and decision makers. DZone offers something for everyone, including news, tutorials, cheat sheets, blogs, feature articles, source code and more. DZone offers something for everyone, including news, tutorials, cheat sheets, blogs, feature articles, source code and more.

DZone communities deliver over 6 million pages each month to more than 3.3 million software developers, architects and decision makers. DZone offers something for everyone, including news, tutorials, cheat sheets, blogs, feature articles, source code and more.

DZone offers something for everyone, including news, tutorials, cheat sheets, blogs, feature articles, source code and more. DZone offers something for everyone, including news, tutorials, cheat sheets, blogs, feature articles, source code and more.