

HTML Advanced: HTML 5



HTML5: New Features

- > Canvas element for drawing
- > Video/audio elements for media playback
- > Better support for local offline storage
- > New content specific elements, like article, footer, header, nav, section
- > New form controls, like calendar, date, time, email, url, search

HTML5: Support

HTML5 is not yet an official standard, and no browser has full HTML5 support.

HTML5: Support

- Question: “How can I start using HTML5 if older browsers don’t support it?”
- Ans: Question itself is misleading.
- HTML5 is not one big thing; it is a collection of individual features. You can only detect support for individual features, like canvas, video, or geolocation.
- 2

HTML5: Support

Love it or hate it, you can't deny that HTML 4 is the most successful markup format ever. HTML5 builds on that success.

You don't need to throw away your existing markup. You don't need to relearn things you already know. If your web application worked yesterday in HTML 4, it will still work today in HTML5. Period.

HTML5: Example

HTML5 supports all the form controls from HTML 4, but it also includes new input controls. Some of these are long-overdue additions like sliders and date pickers; etc.

HTML5: Example

Mobile browsers will customize their onscreen keyboard to make it easier to type email addresses. Older browsers don't support email input type but will treat it as a regular text field, and the form still works with no markup changes or scripting hacks.

```
<!DOCTYPE html>
<html> <body>
<form action="/action_page.php">
  E-mail:
  <input type="email" name="email">
  <input type="submit">
</form>
</body> </html>
```


HTML5: DOCTYPE

The DOCTYPE which comes before the beginning <html> tag is much simpler in HTML 5. Here are some examples of what it looks like now...

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

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

Previous versions of HTML defined a lot of doctypes, and choosing the right one could be tricky. In HTML5, there is only one doctype:

```
<!DOCTYPE html>
```


HTML: Support

Whether you want to draw on a canvas, play video, design better forms, or build web applications that work offline, you'll find that HTML5 is already well-supported.

HTML: Support

Firefox, Safari, Chrome, Opera, and mobile browsers already support canvas, video, geolocation, local storage, and more. Google already supports microdata annotations. Even Microsoft — rarely known for blazing the trail of standards support — will be supporting most HTML5 features in the upcoming Internet Explorer 9.

HTML5 - Microdata

- Standard to provide additional semantics in web pages.
- Lets you define your own customized elements & start embedding custom properties in web pages.
- Consists of a **group** of name-value pairs.
- The groups are called **items**, and each name-value pair is a **property**.
- To create an item, **itemscope** attribute is used.
- To add a property to an item, the **itemprop** attribute is used.
- The following HTML5 markup is of a typical “About” page containing information about a person:

HTML5 - Microdata

- The following HTML5 markup is of a typical “About” page containing information about a person:
- `<section> Hello, my name is A. Rahman, I am security consultant at the Cybernetics Corporation. My friends call me Guru. You can visit my homepage at alphapeeler.sf.net. I live at 1453/14 Liaqat Street, Dastagir society, F.B. Area. </section>`

HTML5 - Microdata

- Same markup with added Schema.org Microdata:

```
<section itemscope itemtype="http://schema.org/Person">
Hello, my name is
<span itemprop="name">A. Rahman</span>,
I am
<span itemprop="jobTitle">security consultant</span>
at the <span itemprop="affiliation">Cybernetics Corporation</span>.
My friends call me
<span itemprop="additionalName">Guru</span>.
You can visit my homepage at
<a href="http://alphapeeler.sf.net" itemprop="url">
alphapeeler.sf.net</a>.
<section itemprop="address" itemscope
itemtype="http://schema.org/PostalAddress">
    I live at
    <span itemprop="streetAddress"> 1453/14 Liaqat Street</span>,
    <span itemprop="addressLocality"> Dastagir society </span>,
    <span itemprop="addressRegion">F.B. Area</span>.
</section>
</section>
```

HTML5: Detection

When your browser renders a web page, it constructs a **Document Object Model**, a collection of objects that represent the HTML elements on the page. Every element is represented in the DOM by a different object.

In browsers that support HTML5 features, certain objects will have unique properties. A quick peek at the DOM will tell you which features are supported.

HTML5: Detection

[Modernizr](#) is an open source, MIT-licensed JavaScript library that detects support for many HTML5 & CSS3 features. To use it, include the following `<script>` element at the top of your page...

HTML5: Detection

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

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

```
  <title>HTML5 sure is fun</title>
```

```
  <script src="modernizr.min.js"></script>
```

```
</head>
```

```
<body>
```

```
  ...
```

```
</body>
```

```
</html>
```

HTML5: Detection

Modernizr runs automatically. When it runs, it creates a global object called Modernizr, that contains a set of Boolean properties for each feature it can detect. For example, if your browser supports the canvas API, the Modernizr.canvas property will be true – otherwise the property will be false.

HTML5: Detection

```
if (Modernizr.canvas) {  
    // let's draw some shapes!  
} else {  
    // no native canvas support available :(  
}
```

More on HTML5 detection (and Modernizr) here:

<http://diveintohtml5.org/detect.html>

HTML5: Video

- Today, most videos are shown through a plugin (like Flash). However, not all browsers have the same plugins.
- HTML5 specifies a standard way to include video with the video element.
- Currently, there are 3 supported video formats for the video element: MP4, Ogg, WebM

HTML5: Video

- IE 8 does not support the video element. In IE 9, there is support for video element using MPEG4.

Browser	MP4	WebM	Ogg
Internet Explorer	YES	NO	NO
Chrome	YES	YES	YES
Firefox	YES from Firefox 21 from Firefox 30 for Linux	YES	YES
Safari	YES	NO	NO
Opera	YES From Opera 25	YES	YES

HTML5: Video

```
<!DOCTYPE HTML>
```

```
<html>
```

```
<body>
```

```
<video src="movie.ogg" width="320" height="240"  
controls="controls">
```

Your browser does not support the video tag.

```
</video>
```

```
</body>
```

```
</html>
```

HTML5: Video

The video element allows multiple source elements. Source elements can link to different video files. The browser will use the first recognized format:

```
<video width="320" height="240" controls="controls">  
  <source src="movie.ogg" type="video/ogg" />  
  <source src="movie.mp4" type="video/mp4" />  
  <source src="movie.webm" type="video/webm" />
```

Your browser does not support the video tag.

```
</video>
```

HTML5: Video

Attribute	Value	Description
<u>autoplay</u>	autoplay	Specifies that the video will start playing as soon as it is ready
<u>controls</u>	controls	Specifies that video controls should be displayed (such as a play/pause button etc).
<u>height</u>	pixels	Sets the height of the video player
<u>loop</u>	loop	Specifies that the video will start over again, every time it is finished
<u>muted</u>	muted	Specifies that the audio output of the video should be muted
<u>poster</u>	URL	Specifies an image to be shown while the video is downloading, or until the user hits the play button
<u>preload</u>	auto metadata none	browser load the entire video when the page loads browser load only metadata when the page loads browser NOT load the video when the page loads
<u>src</u>	URL	Specifies the URL of the video file
<u>width</u>	pixels	Sets the width of the video player

HTML5: Audio

Until now, there has never been a standard for playing audio on a web page.

Today, most audio is played through a plugin (like Flash). However, not all browsers have the same plugins.

HTML5 specifies a standard way to include audio, with the audio element. The audio element can play sound files, or an audio stream.

HTML5: Audio

Currently, there are 3 supported formats for the audio element: Ogg Vorbis, MP3, Wav

Browser	MP3	Wav	Ogg
Internet Explorer	YES	NO	NO
Chrome	YES	YES	YES
Firefox	YES	YES	YES
Safari	YES	YES	NO
Opera	YES	YES	YES

HTML5: Audio

```
<!DOCTYPE HTML>
```

```
<html>
```

```
<body>
```

```
<audio src="song.ogg" controls="controls">
```

Your browser does not support the audio element.

```
</audio>
```

```
</body>
```

```
</html>
```


HTML5: Audio

The audio element allows multiple source elements. Source elements can link to different audio files. The browser will use the first recognized format.

```
<audio controls="controls">
```

```
  <source src="song.ogg" type="audio/ogg" />
```

```
  <source src="song.mp3" type="audio/mpeg" />
```

Your browser does not support the audio element.

```
</audio>
```

HTML5: Audio

Attribute	Value	Description
<u>autoplay</u>	autoplay	Specifies that the audio will start playing as soon as it is ready
<u>controls</u>	controls	Specifies that audio controls should be displayed (such as a play/pause button etc)
<u>loop</u>	loop	Specifies that the audio will start over again, every time it is finished
<u>muted</u>	muted	Specifies that the audio output should be muted
<u>preload</u>	auto	browser load the entire video when the page loads
	metadata	browser load only metadata when the page loads
	none	browser NOT load the video when the page loads
<u>src</u>	URL	Specifies the URL of the audio file.

HTML5: Canvas

The HTML5 canvas element uses JavaScript to draw graphics on a web page.

A canvas is a rectangular area, and you control every pixel of it.

The canvas element has several methods for drawing paths, boxes, circles, characters, and adding images.

HTML5: Canvas

Adding a canvas element to the HTML5 page.

Specify the id, width, height of the element:

```
<canvas id="myCanvas" width="200"  
height="100"></canvas>
```

HTML5: Canvas

The canvas element has no drawing abilities of its own. All drawing must be done inside a JavaScript:

```
<canvas id="myCanvas" width="400" height="350" style="border:1px solid #d3d3dd;">
```

Your browser does not support the HTML5 canvas tag.</canvas>

```
<script>
```

```
var c = document.getElementById("myCanvas");
```

```
var ctx = c.getContext("2d");
```

```
ctx.fillStyle = "#FF0000";
```

```
ctx.fillRect(20, 20, 150, 100);
```

```
</script>
```

HTML5: Canvas

```
<html><body>
```

```
<canvas id="myCanvas" width="300" height="150"  
style="border:1px solid #d3d3d3;">
```

```
Browser does not support HTML5 canvas</canvas>
```

```
<script>
```

```
var c = document.getElementById("myCanvas");
```

```
var ctx = c.getContext("2d");
```

```
ctx.shadowBlur = 20;
```

```
ctx.fillStyle = "red";
```

```
ctx.shadowColor = "blue";
```

```
ctx.fillRect(20, 20, 100, 80);
```

```
</script></body></html>
```




HTML5: Input types

HTML5 has several new input types for forms.

- > email
- > url
- > number
- > range
- > date pickers (date, month, week, time, datetime, datetime-local)
- > search
- > color

HTML5: Input types

Input type	IE	Firefox	Opera	Chrome	Safari
email	No	No	9.0	No	No
url	No	No	9.0	No	No
number	No	No	9.0	7.0	No
range	No	No	9.0	4.0	4.0
Date pickers	No	No	9.0	No	No
search	No	No	11.0	No	No
color	No	No	11.0	No	No

Note: Opera has the best support for the new input types. However, you can already start using them in all major browsers. If they are not supported, they will behave as regular text fields.

HTML5: Input - e-mail

The email type is used for input fields that should contain an e-mail address.

The value of the email field is automatically validated when the form is submitted.

E-mail: `<input type="email" name="user_email" />`

Tip: Safari on the iPhone recognizes the email input type, and changes the on-screen keyboard to match it (adds @ and .com options).

HTML5: Input - url

The url type is used for input fields that should contain a URL address.

The value of the url field is automatically validated when the form is submitted.

Homepage: `<input type="url" name="user_url" />`

Tip: Safari on the iPhone recognizes the url input type, and changes the on-screen keyboard to match it (adds .com option).

HTML5: Input - number

The number type is used for input fields that should contain a numeric value.

Set restrictions on what numbers are accepted:

Points: `<input type="number" name="points" min="1" max="10" />`

Attribute	Value	Description
max	<i>number</i>	Specifies the maximum value allowed
min	<i>number</i>	Specifies the minimum value allowed
step	<i>number</i>	Specifies legal number intervals (if step="3", legal numbers could be -3,0,3,6, etc)
value	<i>number</i>	Specifies the default value

HTML5: Input - range

The range type is used for input fields that should contain a value from a range of numbers.

The range type is displayed as a slider bar.

You can also set restrictions on what numbers are accepted:

```
<input type="range" name="points" min="1" max="10" />
```

Points:

HTML5: Input – date pickers

HTML5 has several new input types for selecting date and time:

- > date - Selects date, month and year
- > month - Selects month and year
- > week - Selects week and year
- > time - Selects time (hour and minute)
- > datetime - Selects time, date, month and year
- > datetime-local - Selects time, date, month and year (local time)

HTML5: Input - search

The search type is used for search fields like a site search or Google search.

The search field behaves like a regular text field.

HTML5: Input – color picker

The color type is used for input fields that should contain a color.

This input type will allow you to select a color from a color picker:

Color: `<input type="color" name="user_color" />`

HTML5: Input - form

HTML5 has several new elements and attributes for forms.

- > datalist
- > keygen
- > output

Attribute	IE	Firefox	Opera	Chrome	Safari
datalist	No	No	9.5	No	No
keygen	No	No	10.5	3.0	No
output	No	No	9.5	No	No

HTML5: Form elements

The datalist element specifies a list of options for an input field. The list is created with option elements inside the datalist.

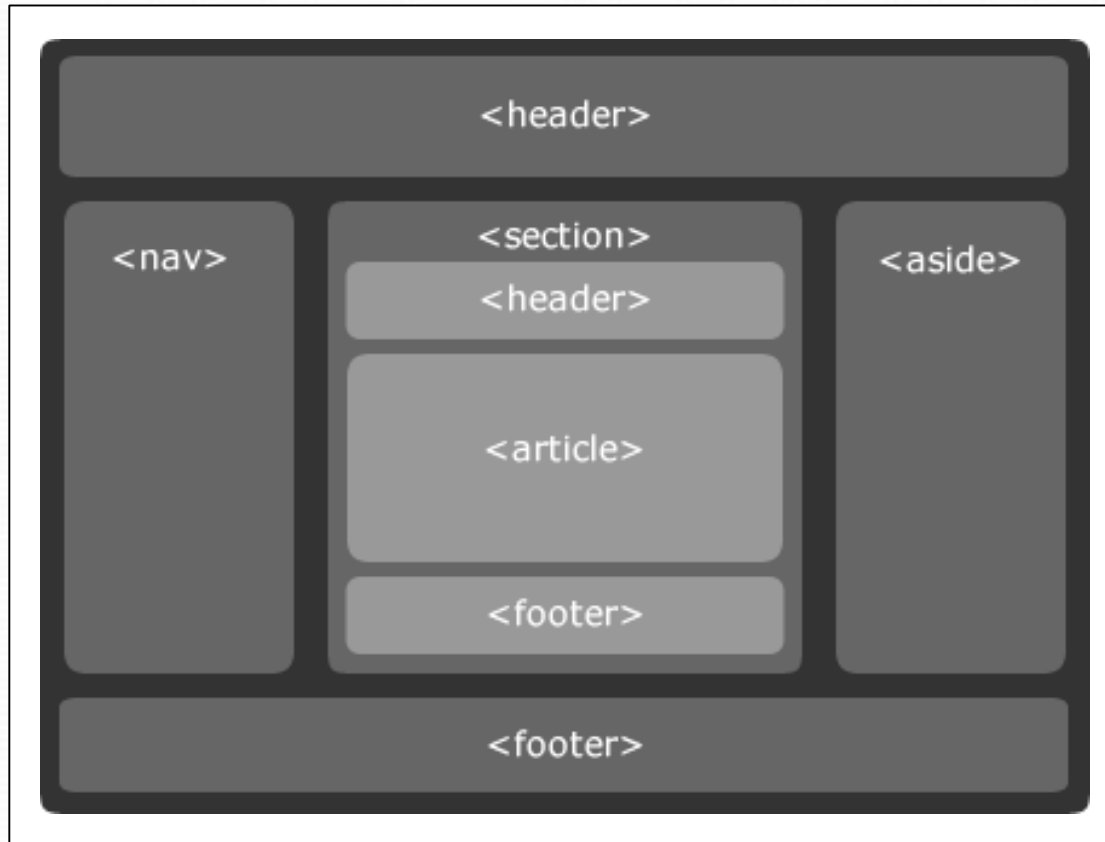
The purpose of the keygen element is to provide a secure way to authenticate users.

The output element is used for different types of output, like calculations or script output:

HTML5: Form attributes

Attribute	IE	Firefox	Opera	Chrome	Safari
autocomplete	8.0	3.5	9.5	3.0	4.0
autofocus	No	No	10.0	3.0	4.0
form	No	No	9.5	No	No
form overrides	No	No	10.5	No	No
height and width	8.0	3.5	9.5	3.0	4.0
list	No	No	9.5	No	No
min, max and step	No	No	9.5	3.0	No
multiple	No	3.5	11.0	3.0	4.0
novalidate	No	No	11.0	No	No
pattern	No	No	9.5	3.0	No
placeholder	No	No	11.0	3.0	3.0
required	No	No	9.5	3.0	No

HTML5: What's New



HTML5 Semantic Elements

Migration from XHTML to HTML5

Typical HTML4

`<div id="header">`

`<div id="menu">`

`<div id="content">`

`<div class="article">`

`<div id="footer">`

Typical HTML5

`<header>`

`<nav>`

`<section>`

`<article>`

`<footer>`

<article>

- The <article> element specifies independent, self-contained content.
- The <section> element defines section in a document.

```
<article>
```

```
  <h1>What is AlphaPeeler?</h1>
```

```
  <p> AlphaPeeler is a crypto educational  
tool. It includes frequency analysis, mono-  
alphabetic substitution, Caesar,  
transposition, Vigenere, & Playfair cipher.  
Professional features are: DES, Gzip, MD5,  
SHA1, SHA256, RIPEMD-16, RSA, & secret  
share files. </p>
```

```
</article>
```

<nav>

- The <nav> element defines a set of navigation links.

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

<figure> and <figcaption>

- 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:

```
<figure>
```

```
  
```

```
  <figcaption>Fig.1 - The Pulpit Rock,  
  Norway.</figcaption>
```

```
</figure>
```



HTML5 Semantic Elements

- Below is an alphabetical list of the new semantic elements in HTML5.

Tag	Description
<u><article></u>	Defines an article
<u><aside></u>	Defines content aside from the page content
<u><details></u>	Defines additional details that the user can view or hide
<u><figcaption></u>	Defines a caption for a <figure> element
<u><figure></u>	Specifies self-contained content, like illustrations, diagrams, photos, code listings, etc.
<u><footer></u>	Defines a footer for a document or section
<u><header></u>	Specifies a header for a document or section
<u><main></u>	Specifies the main content of a document
<u><mark></u>	Defines marked/highlighted text
<u><nav></u>	Defines navigation links
<u><section></u>	Defines a section in a document
<u><summary></u>	Defines a visible heading for a <details> element
<u><time></u>	Defines a date/time

HTML5 - Web Storage

- Cookies disadvantages:
 - Cookies are included with every HTTP request, thereby **slowing down** your web application by transmitting the same data.
 - Cookies are included with every HTTP request, thereby sending data **unencrypted** over the internet.
 - Cookies are **limited to about 4 KB** of data . Not enough to store required data.
- The two storage's are **session storage** and **local storage** in HTML5

Session Storage

- The *Session Storage* is designed for scenarios where the user is carrying out a single transaction, but could be carrying out multiple transactions in different windows at the same time.
- Example: *For example, if a user buying train tickets in two different windows of browser.*
- *Consequences:*
 - *causing the user to buy two tickets for the same flight without really noticing.*

sessionStorage

- HTML5 introduces the *sessionStorage* attribute which would be used by the sites to add data to the session storage, and it will be accessible to any page from the same site opened in that window i.e session and as soon as you close the window, session would be lost.

```
<script type="text/javascript">
    if( sessionStorage.hits ){
        sessionStorage.hits = Number(sessionStorage.hits) +1;
    } else{
        sessionStorage.hits = 1;
    }
    document.write("Total Hits :" + sessionStorage.hits );
</script>
<p>Refresh page to increase hits.</p>
<p>Close window and reopen and check again.</p>
```

Local Storage

- The *Local Storage* is designed for storage that spans **multiple windows**, and lasts beyond the current session. In particular, Web applications may wish to store megabytes of user data, such as entire user-authored documents or a user's mailbox, on the client side for performance reasons.
- Following is the code which would set a local storage variable and access that variable every time this page is accessed, even next time when you open the window:

Local Storage

```
<script type="text/javascript">  
    if( localStorage.hits ){  
        localStorage.hits = Number(localStorage.hits) +1;  
    } else{  
        localStorage.hits = 1; }  
    document.write("Total Hits :" + localStorage.hits );  
</script>  
<p>Refresh page to increase hits.</p>  
<p>Close window, reopen it and check results.</p>
```


Delete Web Storage

- The *Session Storage Data* would be deleted by the browsers immediately after the session gets terminated.
- To clear a local storage setting you would need to call **localStorage.remove('key');** where 'key' is the key of the value you want to remove. If you want to clear all settings, you need to call **localStorage.clear()** method.

Delete Web Storage

```
<script type="text/javascript">  
    localStorage.clear();  
    // Reset number of hits.  
    if( localStorage.hits ){  
        localStorage.hits = Number(localStorage.hits) +1;  
    } else{  
        localStorage.hits = 1;  
    }  
    document.write("Total Hits :" + localStorage.hits );  
</script>  
<p>Refreshing page would not increase hits.</p>  
<p>Close window, reopen it and check results.</p>
```

HTML5 - Web SQL Database

- The Core Methods
- There are following three core methods defined in the spec that I'm going to cover in this tutorial –
- **openDatabase** – This method creates the database object either using existing database or creating new one.
- **transaction** – This method give us the ability to control a transaction and performing either commit or roll-back based on the situation.
- **executeSql** – This method is used to execute actual SQL query.

Opening Database

```
var db = openDatabase('mydb', '1.0', 'Test DB', 2 *  
1024 * 1024);
```

Above method took following four parameters –

- Database name
- Version number
- Text description
- Size of database

Executing queries

- `var db = openDatabase('mydb', '1.0', 'Test DB', 2 * 1024 * 1024);`
- `db.transaction(function (tx) {`
- `tx.executeSql('CREATE TABLE IF NOT EXISTS LOGS (id unique, log)');`
- `});`

INSERT Operation

```
var db = openDatabase('mydb', '1.0', 'Test DB', 2 * 1024 * 1024);
db.transaction(function (tx) {
    tx.executeSql('CREATE TABLE IF NOT EXISTS LOGS (id unique, log)');
    tx.executeSql('INSERT INTO LOGS (id, log) VALUES (1, "foobar")');
    tx.executeSql('INSERT INTO LOGS (id, log) VALUES (2, "logmsg")');
});
```

- We can pass dynamic values while creating entering as follows:

```
var db = openDatabase('mydb', '1.0', 'Test DB', 2 * 1024 * 1024);

db.transaction(function (tx) {
    tx.executeSql('CREATE TABLE IF NOT EXISTS LOGS (id unique, log)');
    tx.executeSql('INSERT INTO LOGS (id,log) VALUES (?, ?)', [e_id, e_log];
});
```


READ Operation

```
var db = openDatabase('mydb', '1.0', 'Test DB', 2 * 1024 * 1024);
```

```
db.transaction(function (tx) {  
    tx.executeSql('CREATE TABLE IF NOT EXISTS LOGS (id unique, log)');  
    tx.executeSql('INSERT INTO LOGS (id, log) VALUES (1, "foobar")');  
    tx.executeSql('INSERT INTO LOGS (id, log) VALUES (2, "logmsg")');  
});
```

```
db.transaction(function (tx) {  
    tx.executeSql('SELECT * FROM LOGS', [], function (tx, results) {  
        var len = results.rows.length, i;  
        msg = "<p>Found rows: " + len + "</p>";  
        document.querySelector('#status').innerHTML += msg;  
  
        for (i = 0; i < len; i++){  
            alert(results.rows.item(i).log );  
        }  
  
    }, null);  
});
```

HTML5 geolocation

- It utilizes the three methods that are packed into the navigator.geolocation object — `getCurrentPosition()`, `watchPosition()` and `clearWatch()`.
- The following is a simple example of geolocation that displays your current position. But, first you need to agree to let the browser tell the web server about your position.

HTML5 geolocation

```
<script type="text/javascript">
  if(navigator.geolocation){
    navigator.geolocation.getCurrentPosition(function(position){
      var positionInfo = "Your current position is (" + "Latitude: " +
position.coords.latitude + ", " + "Longitude: " + position.coords.longitude + ")";
      document.writeln(positionInfo);
    });
  } else{
    alert("Sorry, your browser does not support HTML5 geolocation.");
  }
</script>
```

Showing Location on Google Map

```
<script type="text/javascript">
  navigator.geolocation.getCurrentPosition(showMap);
  function showMap(position){
    var latlong = position.coords.latitude + "," +
position.coords.longitude;
    var mapLink =
"http://maps.googleapis.com/maps/api/staticmap?center="+
latlong+"&zoom=16&size=400x300&output=embed&key=x
xxxxxxxxxxxxxxxxxxxxxxxxxxxxx";
    document.write("<img alt='Map Holder' src='"+
mapLink +"'>");
  }
</script>
```

Google Maps Platform

Credentials

All Google Maps Platform APIs + CREATE CREDENTIALS

Overview

APIs

Metrics

Quotas

Credentials

Support

Solution library NEW

Map management

Map styles

Credentials compatible with this API

To view all credentials visit [Credentials in APIs and services](#)



Remember to configure the OAuth consent screen

API key

Identifies your project using a simple API key to check quota and access

OAuth client ID

Requests user consent so that your app can access the user's data.

Service account

Enables server-to-server, app-level authentication using robot accounts

Help me choose

Asks a few questions to help you decide which type of credential to use

API keys

Name	Creation date	Restrictions ↑	Key	Actions
No API keys to display				

OAuth 2.0 Client IDs

Name	Creation date ↓	Type	Client ID	Actions
No OAuth clients to display				

Service Accounts

[Manage service accounts](#)

Email	Name	Usage with this service (last 30 days) ? ↓	Usage with all services (last 30 days) ?	Actions
No service accounts to display				


API key created

Use this key in your application by passing it with the `key=API_KEY` parameter.

Your API key






~~AKIAIOSFODU7EXAMPLEKEY~~







 Restrict your key to prevent unauthorised use in production.

[CLOSE](#)

[RESTRICT KEY](#)


-  Overview
-  APIs
-  Metrics
-  Quotas
-  Credentials

-  Support
-  Solution library

NEW
-  Map management
-  Map styles





Credentials compatible with this API

To view all credentials visit [Credentials in APIs and services](#)

 Remember to configure the OAuth consent screen with information about your application.

CONFIGURE CONSENT SCREEN

API keys

Name	Creation date	Restrictions	Key	Actions
 API key 1	8 Feb 2022	None	<div></div>	  

OAuth 2.0 Client IDs

Name	Creation date	Type	Client ID	Actions
No OAuth clients to display				

Service Accounts

[Manage service accounts](#)

Email	Name	Usage with this service (last 30 days)	Usage with all services (last 30 days)	Actions
-------	------	--	--	---------