

HTML 5: For Developers

Nathaniel T. Schutta
@ntschorutta

*HTML5 for
Developers
LiveLessons*



[http://www.informit.com/store/
product.aspx?isbn=0132761718](http://www.informit.com/store/product.aspx?isbn=0132761718)



The Plan

- Forms
- Autofocus
- Input types
- Semantic Elements
- Geolocation
- Canvas

Forms.

Forms aren't
interesting.

Right?

HTML5 improves things.

Adds placeholder text!

Very helpful.

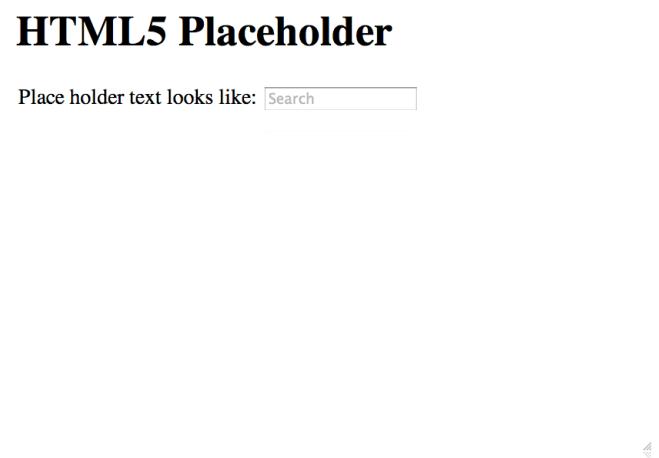
We've seen this before.



So how do we get
in on the action?

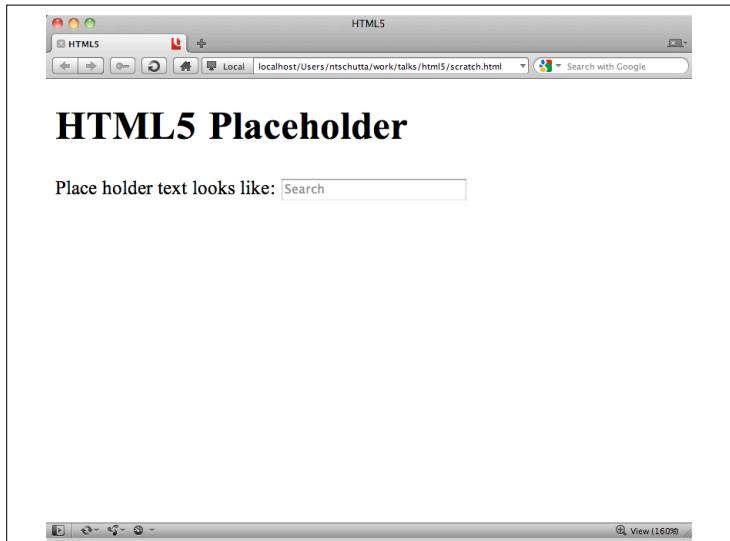
With the placeholder
attribute of course!

If your browser
supports it, you'll see...



```
<!DOCTYPE HTML>
<html>
<head>
  <title>HTML5 Input Types</title>
</head>
<body>
  <h1>HTML5 Adds New Input Types</h1>
  Place holder text looks like: <input type="text" placeholder="Search">
</body>
</html>
```

And if your browser
doesn't support it?



It's ignored.

Can I add markup?

Sorry.

Text only.

Can I style it?

Sort of.

Vendor prefix.

This tests that you can set the placeholder text color.

| | |
|----------|--|
| text | <input type="text" value="search"/> |
| password | <input type="password" value="disabled text"/> |
| default | <input type="text" value="default disabled"/> |

<http://trac.webkit.org/export/37527/trunk/LayoutTests/fast/forms/placeholder-pseudo-style.html>

This tests that you can set the placeholder text color.

| | |
|----------|--|
| text | <input type="text" value="search"/> |
| password | <input type="password" value="disabled text"/> |
| default | <input type="text" value="default disabled"/> |

View (2108)

To change it yourself...

HTML5 Placeholder

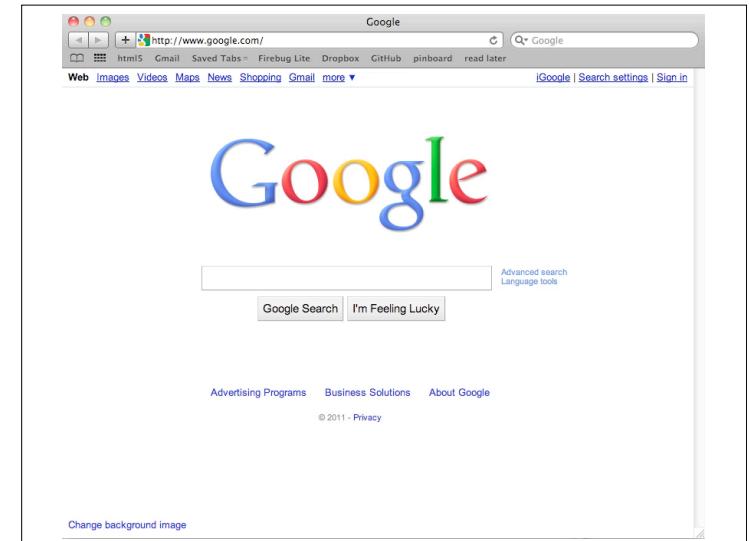
Place holder text looks like:



```
<style>
::-webkit-input-placeholder {
    color: red;
}
</style>
```

Autofocus.

Very handy.



Often done with JavaScript.

**Most of the time,
this is great.**

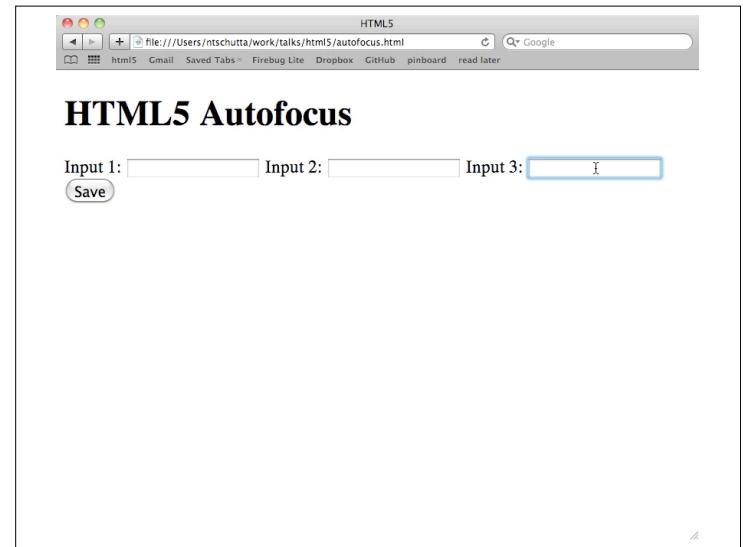
Except when you're
not expecting it...

Helpfully “moves”
your focus.

And you can't opt out.

HTML5 gives us the
autofocus attribute.

After page load, moves
focus to the field.

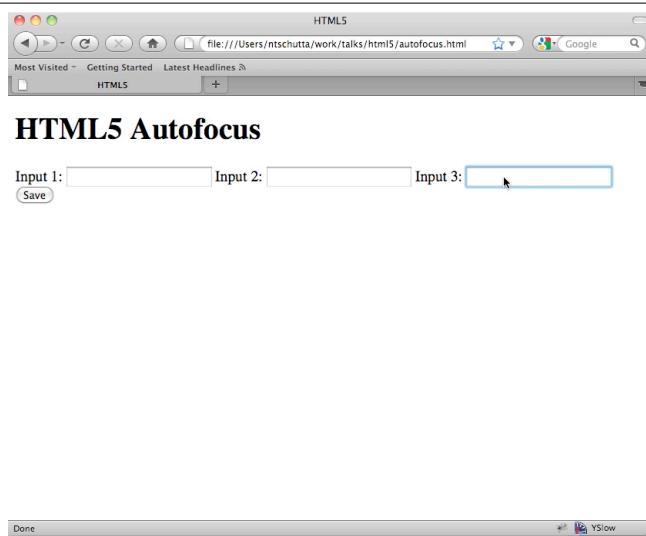


```
<!DOCTYPE HTML>
<html>
<head>
  <title>HTML5</title>
</head>
<body>
  <h1>HTML5 Autofocus</h1>
  <form>
    Input 1: <input type="text" autofocus>
    Input 2: <input type="text">
    Input 3: <input type="text">
    <input type="submit" value="Save">
  </form>
</body>
</html>
```

And if your browser
doesn't support it?

Wait for it...

It's ignored!



So what do you do
about that?

No worries!

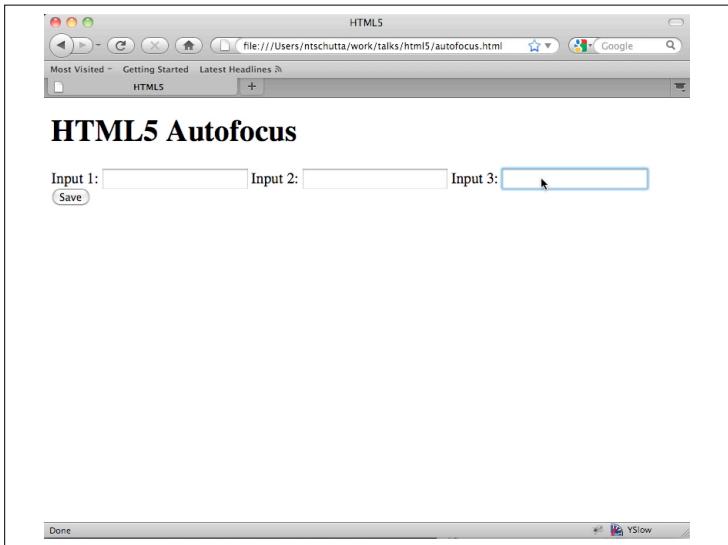
Add the attribute.

Do some feature detection.

Use your own script.

Or rely on a library.

Like say jQuery ;)



```
<script src=jquery-1.5.js></script>
<script>
$(document).ready(function() {
  if (!("autofocus" in document.createElement("input"))) {
    $("#first").focus();
  }
});
```

New Input Types.

We've spent a lot of time developing apps.

With a really limited palette.

Text box, text area, drop down, radio button...

Pretty limited.

Libraries help!

But why doesn't the browser do more?

Now it can!

HTML5 adds 13 new types.

And if your browser doesn't support it?

No worries.

Unknown types
treated as text.

Even works in IE 6!

So what's been added?

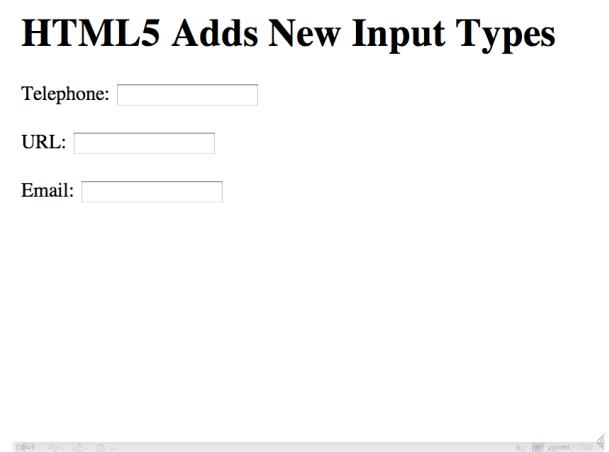
- search
- spinner
- slider
- color picker
- telephone number
- url
- email
- date, month, week, timestamp
- datetime

What do they do?

The spec doesn't say.

In many cases, they look just a text box.

For example...



```
<!DOCTYPE HTML>
<html>
<head>
  <title>HTML5 Input Types</title>
</head>
<body>
  <h1>HTML5 Adds New Input Types</h1>

  Telephone: <input type="tel"> <br/> <br/>
  URL: <input type="url"> <br/> <br/>
  Email: <input type="email"> <br/> <br/>

</body>
</html>
```

Impressed?

Yeah...

So what's the point?

What about the iPhone?

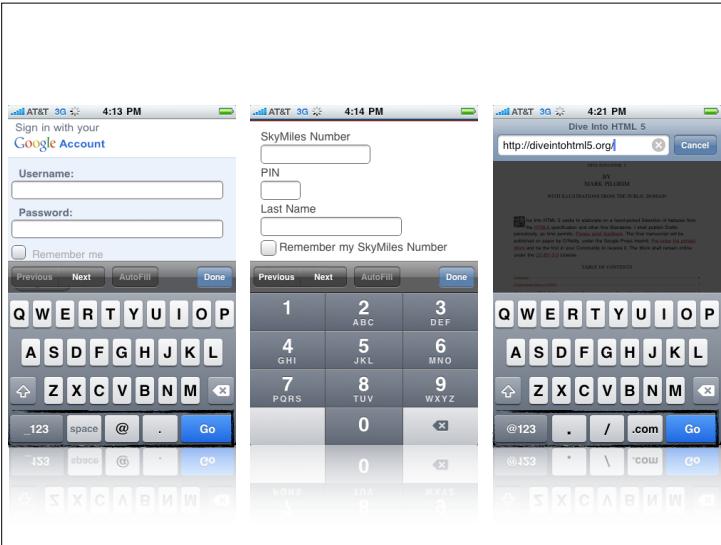
No keyboard.

“Need a keyboard.”

Really?

Can't reconfigure a physical keyboard.

But when it's software...



That's useful!

Frustrating when sites don't.

And it costs you nothing.

Search.

Speaking of inputs that
don't look much different...

HTML5 Adds New Input Types

Search:



```
<!DOCTYPE HTML>
<html>
<head>
    <title>HTML5 Input Types</title>
</head>
<body>
    <h1>HTML5 Adds New Input Types</h1>
    Search: <input type="search" autofocus> <br/><br/>
</body>
</html>
```

Rounded corners!

Oh, and an X...



HTML5 Adds New Input Types

Search:

Numbers.

Like email addresses &
URLs, they're special.

How about spinners
and ranges?



HTML5 Adds New Input Types

Number:

Range:

```
<!DOCTYPE HTML>
<html>
<head>
  <title>HTML5 Input Types</title>
</head>
<body>
  <h1>HTML5 Adds New Input Types</h1>

  Number: <input type="number"
    step="2"
    value="0"
    min="0"
    max="10"> <br/> <br/>

  Range: <input type="range"> <br/> <br/>

</body>
</html>
```

Attributes are optional.

Default step value is 1.

Ranges.

Slider control.

Shades of thick clients!

```
<!DOCTYPE HTML>
<html>
<head>
    <title>HTML5 Input Types</title>
</head>
<body>
    <h1>HTML5 Adds New Input Types</h1>

    Number: <input type="number"
                    step="2"
                    value="0"
                    min="0"
                    max="10"> <br/> <br/>
```

```
Range: <input type="range"> <br/> <br/>
</body>
</html>
```

Also includes some
handy JavaScript.

`stepUp(n)`
`stepDown(n)`

Increase/decrease
the value by n.

`valueAsNumber`

Returns value
as a number!

The value attribute
is a string...

Useful?

Back to the iPhone...



Date pickers.

Why don't we have a native date picker?

Now we do!

But it's only in Opera.

And you may not like it.

The screenshot shows the Opera browser interface with a title bar "HTML5 Input Types". The main content area displays a form titled "HTML5 Adds New Input Types" containing the following fields:

- Search:
- Search (Placeholder):
- Number: (value: 0)
- Range:
- Color:
- Telephone:
- URL:
- Email:
- Date:
- Month:
- Week:
- Time:
- Date/Time:

```
<!DOCTYPE HTML>
<html>
<head>
    <title>HTML5 Input Types</title>
</head>
<body>
    <h1>HTML5 Adds New Input Types</h1>

    Date: <input type="date"> <br/> <br/>
    Month: <input type="month"> <br/> <br/>
    Week: <input type="week"> <br/> <br/>
    Time: <input type="time"> <br/> <br/>
    Date/Time: <input type="datetime"> <br/> <br/>

</body>
</html>
```

The screenshot shows the Opera browser interface with a title bar "Date Example - Ajax (jQuery)". The main content area displays a form titled "Really Simple Address Book" with the following fields:

| | |
|-----------|--|
| Name: | <input type="text"/> |
| Email: | <input type="text"/> |
| Phone: | <input type="text"/> |
| Street: | <input type="text"/> |
| City: | <input type="text"/> State: <input type="text"/> Zip: <input type="text"/> |
| Website: | <input type="text"/> |
| IM: | <input type="text"/> |
| Birthday: | <input type="text"/> |

CSS could be improved.

**But which would
your users' prefer?**

Fallback to a library.

Speaking of pickers.

Color!

**Pick a color, get
a hex value!**

Cool!

Only works in Opera.

Bummer.

Uses native picker.

Except on Linux.



HTML5 Adds New Input Types

Search:

```
<!DOCTYPE HTML>
<html>
<head>
  <title>HTML5 Input Types</title>
</head>
<body>
  <h1>HTML5 Adds New Input Types</h1>

  Color Picker: <input type="color">

</body>
</html>
```

Validation!

Email address.

**Yes, you can do this
in JavaScript today.**

Maybe you already do.

It's hard!

What if JS is disabled?

**And you're still validating
on the server right?**

HTML5 to the rescue!

A screenshot of a web browser window titled "HTML5 Input Types". The address bar shows "localhost/Users/ntschutta/work/talks/html5/scratch.html?r=1". The main content area displays the text "HTML5 Adds New Input Types" followed by a form with a label "Email:" and a blue-bordered input field. Below the input field is a "Submit" button. At the bottom of the browser window, there is a toolbar with icons and a status bar showing "View (1590)".

HTML5 Adds New Input Types

Email:

Submit

Validation is on by default.

**Also works for url
and numbers.**

Even respects min/max.

Don't want to validate?

Use novalidate attribute.

Support is...soft.

**Safari and Chrome,
no error messages.**

Just doesn't submit ;)

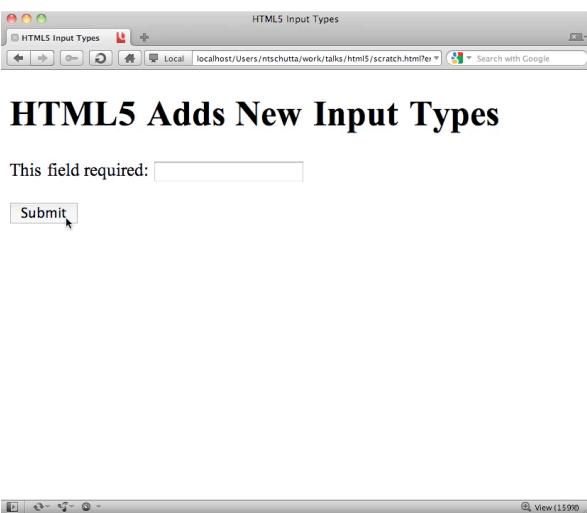
Very user friendly.

Required fields.

Add the required attribute!

Appearance varies
by browser.

For example...



```
<!DOCTYPE HTML>
<html>
<head>
  <title>HTML5 Input Types</title>
</head>
<body>
  <h1>HTML5 Adds New Input Types</h1>
  <form>
    This field required: <input name="foo" required>
    <input type="submit" value="Submit">
  </form>
</body>
</html>
```

New semantic elements.

HTML5 adds new elements.

- section
- nav
- article
- aside
- hgroup
- header
- footer
- time
- mark

Defines things we've
been doing for years.

With divs and ids.

It works, but
lacks meaning.

Common markup.

Again, nod to what
we're actually doing.

More meaningful than divs!

So what do these
elements mean?

<section>

Thematic grouping
of content.

Might have heading
or an outline.

Chapters, tabs.

Intro, part I, part
2...part N, conclusion.

<nav>

Section with links.

Major navigation blocks.

Common in footers.



Nothing tells you that's navigation though.

Common yes...

Accessibility.

Screen readers, keyboard only users.

`<article>`

**Reusable or
distributable.**

Post, blog entry, comment.

Think syndication.

`<aside>`

Tangential content.

Sidebars, pull quotes.

<hgroup>

Group of set of
headings (h1-h6).

<header>

Introduction.

Could contain hgroup
or headings.

**Does't create a
new section.**

**Not a new scope for
headers/footers.**

```
<div id="header">  
...  
</div>
```

```
<header>  
...  
</header>
```

```
<footer>
```

**Usually at the bottom
of a section.**

Often contains copyright,
contact info, help, privacy, etc.

Whatever lives in the
div id="footer";)

Does't create a
new section.

<time>

Encode time/date for
machine use.

Meetings, birthdays,
anniversaries ;)

```
<time datetime="2011-02-22" pubdate>February 22, 2011</time>
```

3 parts.

I. Machine readable.

```
<time datetime="2011-02-22" pubdate>February 22, 2011</time>
```

YYYY-MM-DD

Quite flexible.

<http://www.whatwg.org/specs/web-apps/current-work/multipage/common-microsyntaxes.html#valid-global-date-and-time-string>

Want time?

Add T, time in 24 hour,
timezone offset.

`datetime="2011-02-22T11:21:37-07:00"`

2. Human readable.

`<time datetime="2011-02-22" pubdate>February 22, 2011</time>`

Text doesn't have to match
the datetime attribute.

It's human readable!

**Next Sunday, tomorrow,
in three days...**

Could even be empty.

pubdate flag.

```
<time datetim="2011-02-22" pubdate>February 22, 2011</time>
```

Boolean.

Says timestamp is publication date.

For article or the document...

<mark>

Think highlight.

Call attention to something.

And if your browser doesn't support it?

Unknown elements
rendered inline.

However, many of these
elements are block.

In older browsers...

Style them as block.

HTML5 Reset.

Oh, before 9, IE won't
style unknown elements.

<http://html5doctor.com/html-5-resetstylesheet/>

Despite your CSS.

Also affects the DOM.

The workaround?

Create the element
in JavaScript.

IE will allow
you to style it.

Don't want to do
that yourself?

No worries.

HTML5 enabling script.

<http://remysharp.com/2009/01/07/html5-enabling-script/>

What's all the fuss about?

Divs work, right?

Document outline.

<http://gsnedders.html5.org/outliner/>

Before, headings were
our only hope.

**Sectioning content (article,
aside, nav, section)...**

Create new nodes.

**Each has its
own hierarchy.**

Aids composability.

Geolocation.

**Where in the
world are you?**



Very helpful on phones!

Technically not a part of HTML5.

Geolocation Working Group.

Wide browser support.

Your browser doesn't support it?

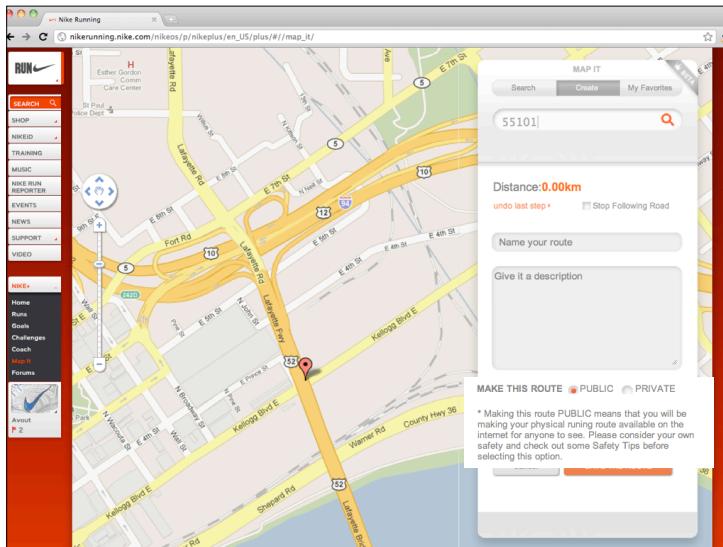
Device specific options.

Privacy issue?

Absolutely.

If you know where
the device is...

<http://icanstalku.com/why.php>



Opt-in.

<http://www.w3.org/TR/geolocation-API/#security>

Browsers tell you
before data is sent.

HTML5 Adds Geolocation

[Where am I?](#)

HTML5 Adds Geolocation

[Where am I?](#)

Location-Aware **BROWSING**

Firefox can tell websites where you're located so you can find info that's more relevant and more useful. It's about making the Web smarter – and is done in a way that totally respects your privacy. [Give it a try!](#)



Frequently asked **QUESTIONS**

+ What is Location-Aware Browsing?

+ How does it work?

Infobars are smart.

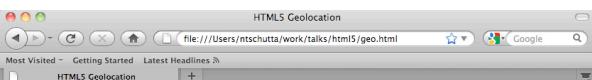
Often give link to
further information.

Aren't modal.

Tab specific.

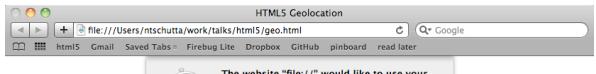
Blocks.

You can go about your business in other tabs.



HTML5 Adds Geolocation

How does this work?



HTML5 Geolocation

Where am I?

```
function whereAmI() {
  if (Modernizr.geolocation) {
    navigator.geolocation.getCurrentPosition(showLocation);
  } else {
    alert("this browser doesn't support geolocation, sorry!");
  }
}

function showLocation(position) {
  var latitude = position.coords.latitude;
  var longitude = position.coords.longitude;
  $("#location").html("lat: " + latitude + " and long: " + longitude);
}
```

New property.

navigator.geolocation

Simple API.

getCurrentPosition()

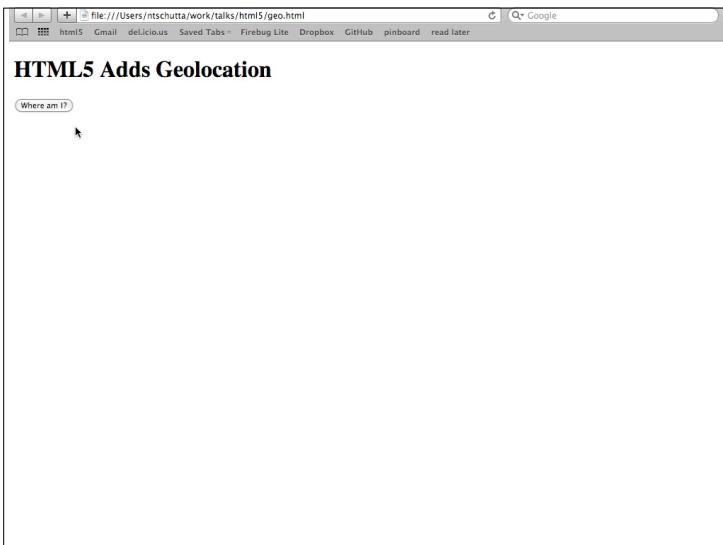
Browser “determines location,” creates Position.

Position contains Coordinates & timestamp.

Coordinates

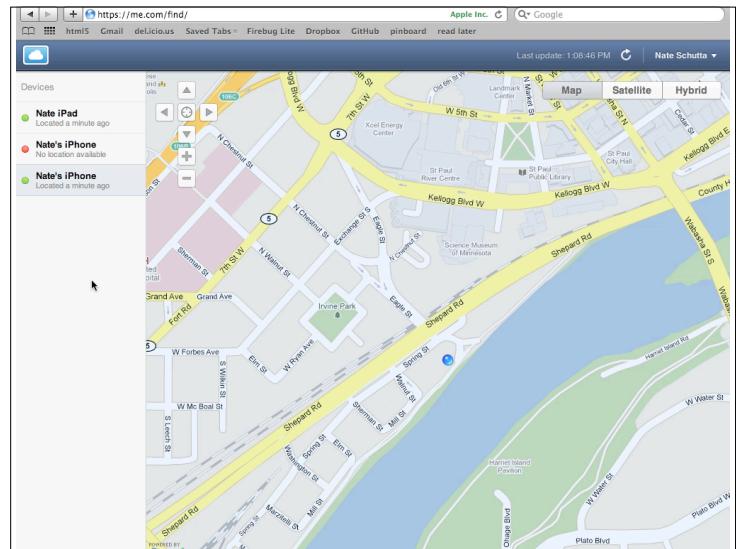
- latitude (double)
- longitude (double)
- altitude (double or null)
- accuracy (double)
- altitudeAccuracy (double or null)
- heading (double or null)
- speed (double or null)

You supply a callback function.



How accurate is it?

Can be ***very*** accurate.



Your function receives
a Position object with:

coords & timestamp

Can take some time ;)

`getCurrentPosition()`

Optional second arg:
error callback function.

HTML5 Adds Geolocation

[Where am I?](#)

```
function whereAmI() {
  if (Modernizr.geolocation) {
    navigator.geolocation.getCurrentPosition(showLocation, handleError);
  } else {
    alert("this browser doesn't support geolocation, sorry!");
  }
}

function showLocation(position) {
  var latitude = position.coords.latitude;
  var longitude = position.coords.longitude;
  $("#location").html("lat: " + latitude + " and long: " + longitude);
}

function handleError(error) {
  if(error.code == 1) {
    var message = "You want to keep your location private, that's OK!"
  }
  $("#location").html("Ooops! " + message);
}
```

Callback gets a
PositionError object.

Two attributes.

code & message

Code values...

- PERMISSION_DENIED (1)
- POSITION_UNAVAILABLE (2)
- TIMEOUT (3)
- UNKNOWN_ERROR (0)

`getCurrentPosition()`

Optional third argument.

PositionOptions

- enableHighAccuracy (boolean)
- timeout (long)
- maximumAge (long)

All attributes are optional.

Higher accuracy
may be slower.

Some devices have separate permissions.

Timeout is based on network time, not user.

Age allows you to cache positions.

IE? Out of luck < 9.

Are other options.

Gears, device specific.

geo.js

<http://code.google.com/p/geo-location-javascript/>

**Layer over various
approaches.**

Canvas.

Graphics!

**Graphs, shapes,
animations, etc.**

Controlled via scripting.

Pretty simple.

```
<canvas id="canvas" width="800" height="800"></canvas>
```

That's it?

Only two attributes:
width and height.

Optional, default is
300 pixels by 150 pixels.

CSS sizing as well.

Can be styled like an image
- border, margin, etc.

Specifying fallback content.

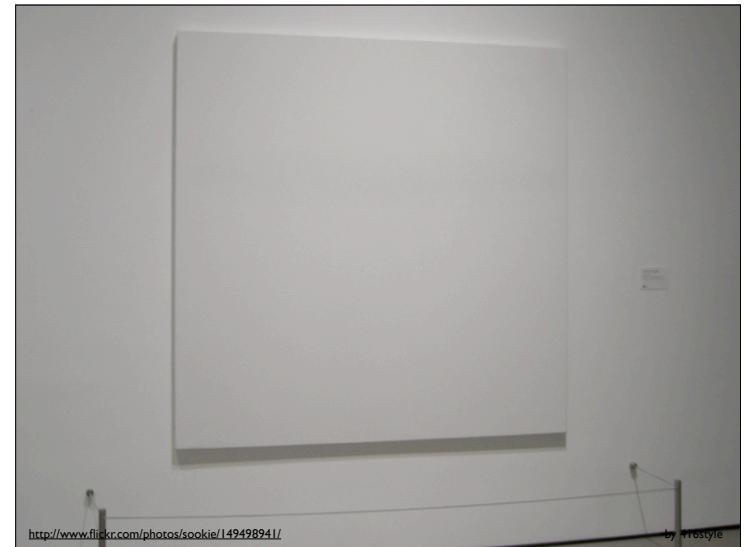
```
<canvas id="fallback" width="800" height="800">  
  Put fallback content here...perhaps an image.  
</canvas>
```

Content between tag.

**Ignored by browsers
supporting canvas...**

**Canvas tag is ignored by
browsers lacking support.**

Canvas starts like
any canvas...



Up to you to fill it!

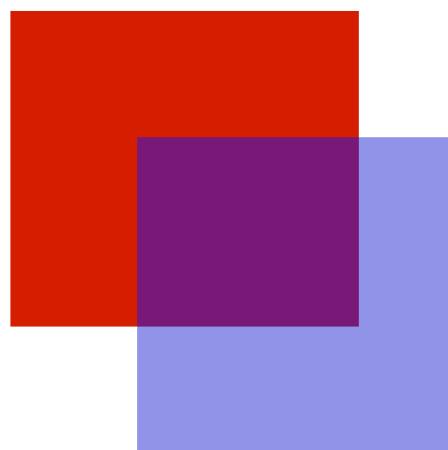
To draw, we need a context.

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

For now, just 2D...

Likely 3D in the future.

Once we have a context, we can draw!



```
<head>
<script type="application/javascript">
function draw() {
    var canvas = document.getElementById("canvas");
    var ctx = canvas.getContext("2d");

    ctx.fillStyle = "rgb(200,0,0)";
    ctx.fillRect(100, 100, 550, 500);

    ctx.fillStyle = "rgba(0, 0, 200, 0.5)";
    ctx.fillRect(300, 300, 550, 500);
}
</script>
</head>
<body onload="draw()">
<canvas id="canvas" width="800" height="800"></canvas>

<canvas id="fallback" width="800" height="800">
    Put fallback content here...perhaps an image.
</canvas>
```

One primitive - rectangle.

Three methods.

```
fillRect(x, y, width, height);  
strokeRect(x, y, width, height);  
clearRect(x, y, width, height);
```

All take the same
arguments...

x and y position of left
corner of rectangle...

Width and height.

fillRect - filled rectangle.

strokeRect - outline.

`clearRect` - clears area,
makes it transparent.

So that's it? Rectangles?

No!

Paths.

We can draw shapes.

- `beginPath` - creates path
- `closePath` - tries to close the shape
- `stroke` - draws an outlined shape
- `fill` - draws a solid shape
- `moveTo` - moves the “pen”, doesn’t draw



[https://developer.mozilla.org/en/
Canvas_tutorial%3aDrawing_shapes](https://developer.mozilla.org/en/Canvas_tutorial%3aDrawing_shapes)

```
function draw() {  
    var canvas = document.getElementById("canvas");  
    var ctx = canvas.getContext("2d");  
  
    ctx.beginPath();  
    ctx.arc(75,75,10,0,Math.PI*2,true); // Outer circle  
    ctx.moveTo(110,75);  
    ctx.arc(75,75,35,0,Math.PI,true); // Mouth (clockwise)  
    ctx.moveTo(65,65);  
    ctx.arc(60,65,5,0,Math.PI*2,true); // Left eye  
    ctx.moveTo(95,65);  
    ctx.arc(90,65,5,0,Math.PI*2,true); // Right eye  
    ctx.stroke();  
}
```

Arc? Draws circles.

- x
- y
- radius
- startAngle - start point, radians
- endAngle - end point, radians
- anticlockwise - boolean, clockwise or not

`lineTo(x, y)` - Straight lines

Curves.

`quadraticCurveTo,`
`bezierCurveTo`

And of course you can
combine these...

You can also use images.

Get an image - from
page or from scratch.

`drawImage(image, x, y)`

Useful as backdrops...

Can also scale images.

Add width and height.

You can also crop...

And on and on!

Colors, gradients, line
styles, patterns...

Rotating, scaling,
transforms, compositing.

Animations!

Whew!

We've barely scratched
the surface!

Lots more...

Great time to
get started!

*HTML5 for
Developers
LiveLessons*



[http://www.informit.com/store/
product.aspx?isbn=0132761718](http://www.informit.com/store/product.aspx?isbn=0132761718)

Questions??

Thanks!

Please complete your surveys.