Importance of Performance

Why performance

- A slow site means that you will have less user engagement.

Example

let's say you accidentally, introduce a performance issue, that causes your page to take an extra half second to load. That's just 500 milliseconds.

That's a pretty small difference.

What kind of difference, in user engagement ,do you think this would cause?

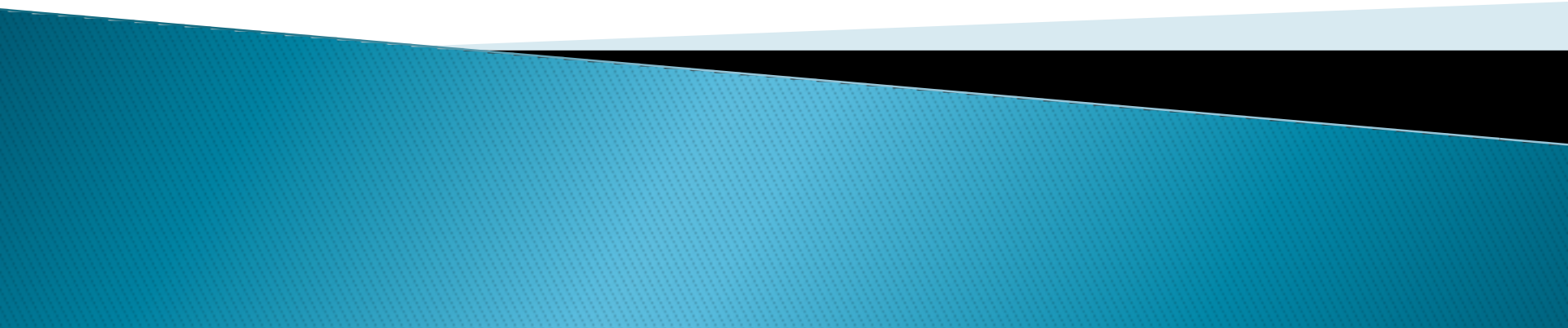
Even such a small change, causes a slightly more than one percent drop

in user engagement, and income from your site.

It is a requirement

Many well publicized studies from Google, Microsoft, Amazon, Facebook, and others, all show that UI performance translates directly to profit. A drop in performance directly drives a drop in user engagement. In short, speed is not a feature, it's a requirement. Users won't use your app if it's not fast.

Analyzing Performance



When to consider Performance

Of course, it's nearly impossible to graph performance into your application after it's already been built, but before we get into good performance patterns, the first step to insuring great performance, is being able to analyze your performance

#perfmatters

Chrome Development Tools

Chrome Development Tools

Understanding Network

Pillars of Performance

- Three pillars of performance are

- Network
- Render
- Compute

Network

- Examples
  1. www.vu.edu.pk
  2. http://playbiolab.com/
  3. https://www.eecs.mit.edu/

Touch Interaction

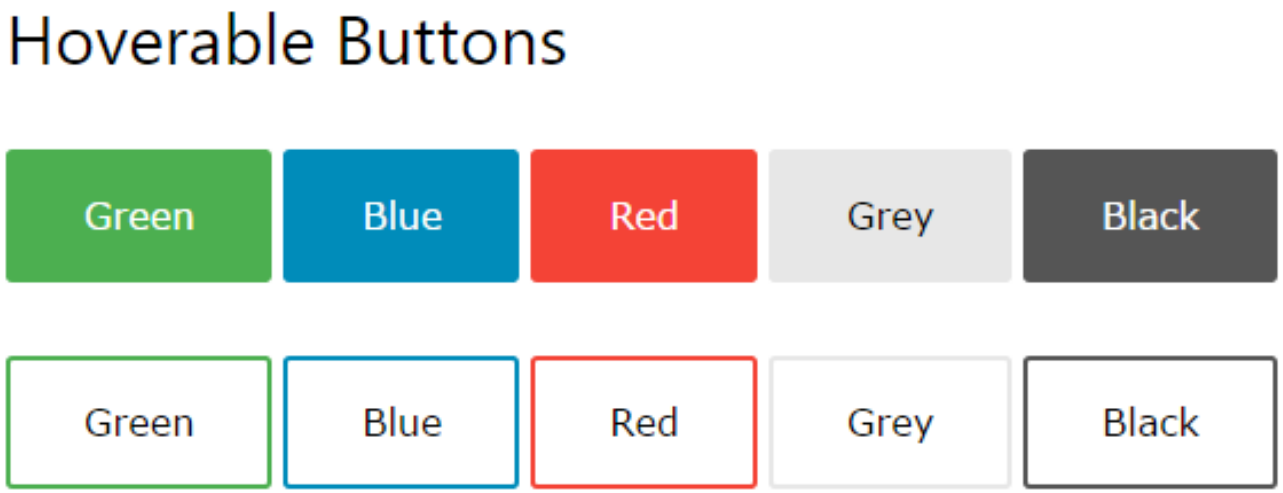
One of the most important factors in an immersive user experience on a mobile device is touch interaction. You expect a responsive interaction from a device that you hold in the palm of your hand, and with most mobile screens, your primary pointing device is your finger.

As a mobile web developer, you need to be sure that you're building great touch enabled user devices

Simple rules for touch UI

Rule 1

Do not relay on Hover Effect



Buttons Source: http://www.w3schools.com/css/css3_buttons.asp

Rule 2

Size of Touch Area should be at-least

10mm X 10 mm

Rule 3

Touch and mouse are not mutually exclusive

Rule 4

Provide Mouse support as well.

Don’t do anything

It is important to understand that you may not even need to do anything in order to support touch.

Touch events emulate mouse clicks already, and the web browser builds in some gestures like scrolling and zooming.

So, you only need to implement something here in cases where the gesture support Or mouse emulation does not give an optimal user experience.

Example

- http://www.w3schools.com/css/css3_buttons.asp

Click Delay

Touch supporting mobile browsers, usually interpret double tapping, as a zoom into this element gesture. Unfortunately, this means the platform, delays firing click of it until it can decide whether, the user is double tapping.

This delay is about a third of a second.

Chrome has recently checked in some changes, that minimize this behavior.

But, pretty much, all mobile browsers,

have this click delay feature, to some degree.

Fix

There are a few ways to fix this problem, and, get clicks without a delay,

1) You can set the view port, to be non-scalable. Either, set user scalable, to no, or set minimum scale and maximum scale, to one, which has the same effect. This can cause accessibility problems, though, so be careful jumping, to this solution. You need to make sure, if you do this, that your site will never ,need to be zoomed.

2) You can use a fast click library, like the ft labs one, that we've linked down below. But ,you do have to be careful, about how this impacts, your scrolling performance, be sure to read the directions, very carefully.

https://github.com/ftlabs/fastclick

3) Where finally, of course, you can implement tech support yourself directly, if you consume the touch events, you won't have this issue.

Supporting Touch

Go to www.caniuse.com

And search for “touch”

Or go to http://caniuse.com/#feat=touch

Touch Events

- touchstart -> mousedown
- touchmove -> mousemove
- touchend -> mouseup
- touchcancel

Touch Events

```
interface TouchEvent : UIEvent {
        readonly    attribute TouchList touches;
        readonly    attribute TouchList targetTouches;
        readonly    attribute TouchList changedTouches;
        readonly    attribute boolean   altKey;
        readonly    attribute boolean   metaKey;
        readonly    attribute boolean   ctrlKey;
        readonly    attribute boolean   shiftKey;
    };
```

Source: https://www.w3.org/TR/touch-events/#touchevent-interface

```
interface Touch {
        readonly    attribute long         identifier;
        readonly    attribute EventTarget target;
        readonly    attribute long         screenX;
        readonly    attribute long         screenY;
        readonly    attribute long         clientX;
        readonly    attribute long         clientY;
        readonly    attribute long         pageX;
        readonly    attribute long         pageY;
  };
```

Source: https://www.w3.org/TR/touch-events/#touch-interface

Touch Events are Fired on The Original Target

Touch events are always delivered to the element that first received that Touch.

They don't walk across boundaries like Mouse Events do. So it's important to hook the right elements. In fact, it's a good idea to add the touch end handler during your touch start handler. And, keep in mind too that even if you remove a DOM element from your tree, it still gets the events until the touch ends.

DOM http://www.w3schools.com/js/js_htmldom.asp

Mouse Emulation

Example:

https://webaudiodemos.appspot.com/midi-synth/


Touch Events
Web Audio MIDI Synthesi
https://webaudiodemos.appspot.com/midi-synth/index.html
MOD
SHAPE
sine
FREQ
OSC1 TREMOLO
OSC2 TREMOLO
OSC1
WAVEFORM
saw
INTERVAL
32'
DETUNE
MIX
OSC2
WAVEFORM
saw
INTERVAL
16'
DETUNE
MIX
FILTER
CUTOFF
Q
MOD
ENV
FILTER ENVELOPE
ATTACK
DECAY
SUSTAIN
RELEASE
VOLUME ENVELOPE
ATTACK
DECAY
SUSTAIN
RELEASE
MASTER
DRIVE
REVERB
VOLUME
MIDI_IN
KBD_OCT
norma
Fork my code on GitHub

Synthesizer Application

Application Link

https://webaudiodemos.appspot.com/

Source Code Download Link

https://github.com/cwilso/midi-synth

We will Initially start with touch events.

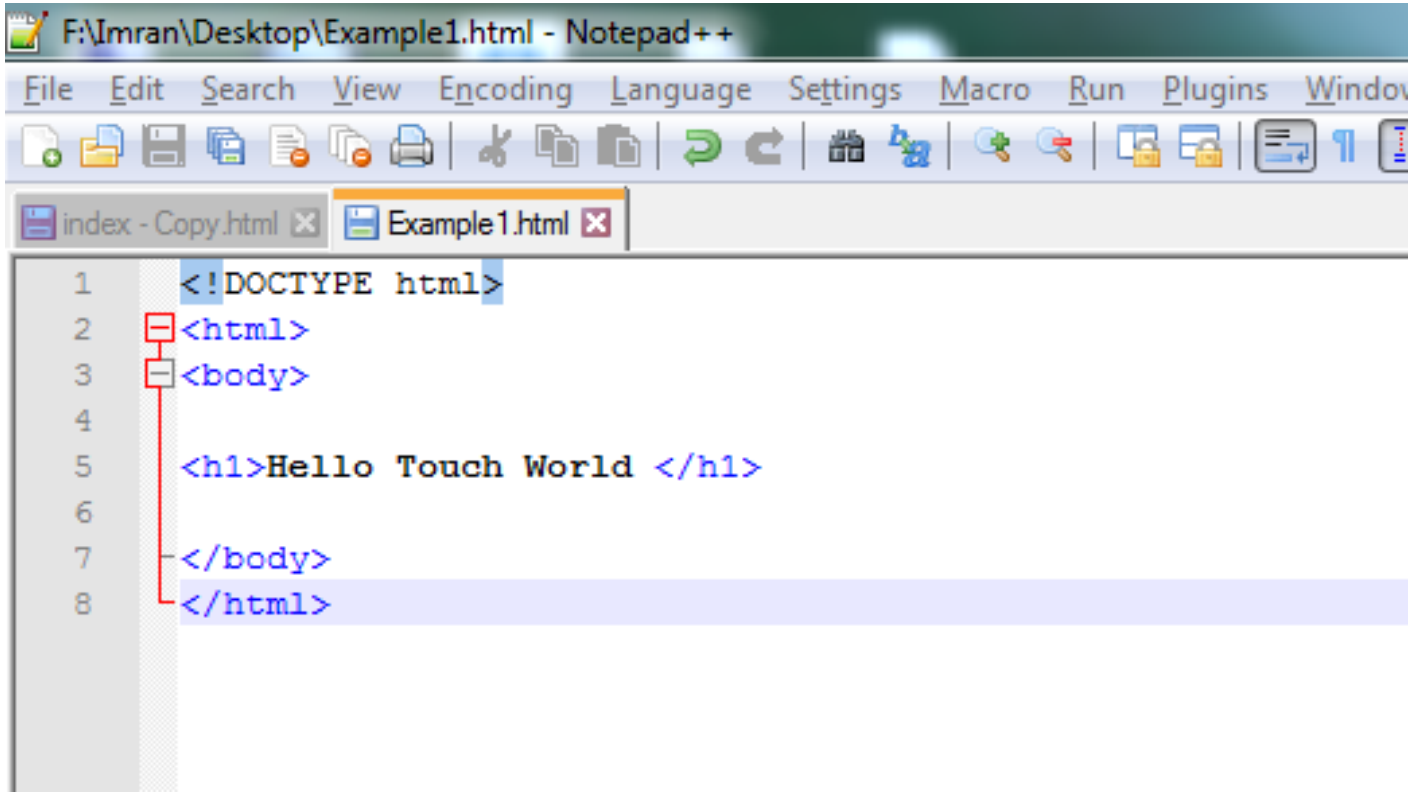
There were two challenges, though. First, you'll notice that on the desktop, if I don't have that touch event emulation turned on, it doesn't actually work at all when I use the mouse.

Of course, this is expected. But I didn't really want two code paths, one for mouse and one for touch. Secondly, I really wanted to be able to have a drag across the keyboard. Play each note as you slid across it in turn. But with the way touch delivers all the events to the originally touched down element, I would have had to calculate the hit testing myself for these. And that was kind of a pain.

I really wanted these touch events to be delivered to the down element for each key. Instead what happens in this synth, is when I hit one key and then drag, it doesn't actually move where the events are delivered.

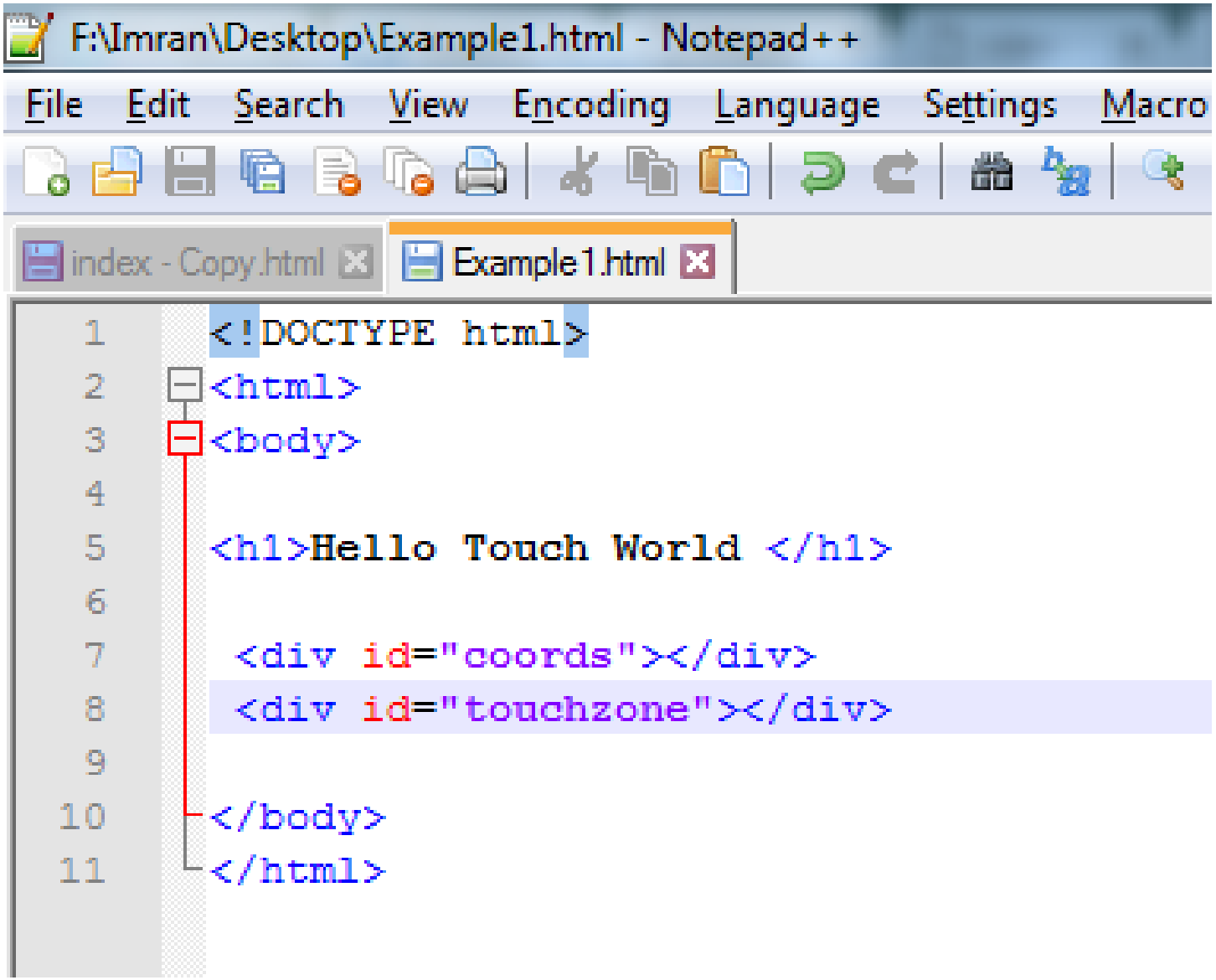
Getting Started With Touch Events

Example – 1


F:\Imran\Desktop\Example1.html - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Windo
index - Copy.html
Example1.html
1 <!DOCTYPE html>
2 <html>
3 <body>
4
5 <h1>Hello Touch World </h1>
6
7 </body>
8 </html>

- We define a *touch-sensitive* div to which we will attach an event listener. We also define a div at the top where we will display the screen coordinates of the most recent touch.

```
1    <div id="coords"></div>
2    <div id="touchzone"></div>
```


F:\Imran\Desktop\Example1.html - Notepad++
File Edit Search View Encoding Language Settings Macro
index - Copy.html
Example1.html
1 <!DOCTYPE html>
2 <html>
3 <body>
4
5 <h1>Hello Touch World </h1>
6
7
8
9
10 </body>
11 </html>

- Next we add some JavaScript to register the event listener, and add a handler function to do something with the touch data:

- So, here we're simply identifying the touchzone div, attaching a listener for the touchstartevent, and registering a handler function: touchHandler.

```
1 function init() {
2   // Get a reference to our touch-sensitive element
3   var touchzone = document.getElementById("touchzone");
4   // Add an event handler for the touchstart event
5   touchzone.addEventListener("touchstart", touchHandler, false);
6 }
```

In the touchHandler function we grab the x and y coordinates of the touch, and write them to the coords div:

```
1 function touchHandler(event) {
2   // Get a reference to our coordinates div
3   var coords = document.getElementById("coords");
4   // Write the coordinates of the touch to the div
5   coords.innerHTML = 'x: ' + event.touches[0].pageX + ', y: ' + event.touches[0].pageY;
6 }
```

Adding Mouse Support

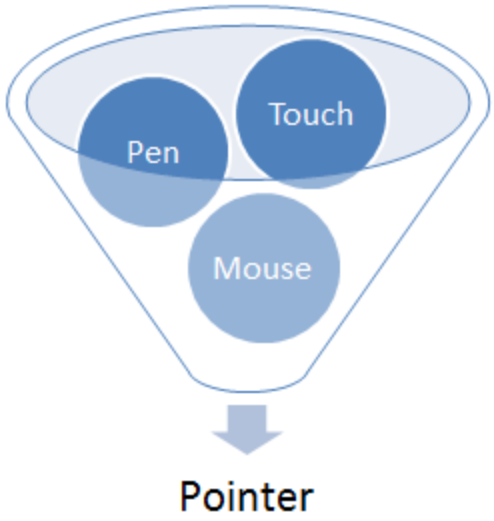
Pointer Events

Problem

To handle the mouse events, touch events and pen events we have to write the code separately for each of these.

Solution

To rectify this situation, Microsoft made a standards proposal that reconciles touch events and mouse events and pen input too, into one model called ***pointer events***.



Support in Browsers

Go to www.caniuse.com

Pointer Events Object

While the Touch Events API was defined in terms of *Touch*es, the Pointer Events API is defined in terms of *Pointer*s,

where a Pointer is defined as:

A hardware agnostic representation of input devices that can target a specific coordinate (or set of coordinates) on a screen

PointerEvent inherits and extends MouseEvent, so it has all the usual properties that MouseEvent has, such as clientX, clientY etc., as well as a few additional ones, such as tiltX, tiltY, and pressure

The pointer attributes

Attribute	Description
pointerId	unique numeric identifier
screenX	horizontal coordinate relative to screen
screenY	vertical coordinate relative to screen
clientX	horizontal coordinate of point relative to viewport, excluding scroll offset
clientY	vertical coordinate of point relative to viewport, excluding scroll offset
pageX	horizontal coordinate of point relative to page, including scroll offset
pageY	vertical coordinate of point relative to page, including scroll offset
width	width of pointer contact on screen
height	height of pointer contact on screen
tiltX	angle of tilt of stylus between Z and X axes, where X and Y plane is on the surface of the screen
tiltY	angle of tilt of stylus between Z and Y axes, where X and Y plane is on the surface of the screen
pressure	pressure of contact on screen
pointerType	class of Pointer: mouse, pen, or touch
isPrimary	is this the main Pointer for a pointer type

Event types defined by the **PointerEvent** interface

event type	fired when…
pointerover	pointer moves over an element (enters its hit test boundaries)
pointerenter	pointer moves over an element or one of its descendants. Differs to pointerover in that it doesn't bubble
pointerdown	active buttons state is entered: for touch and stylus, this is when contact is made with screen; for mouse, when a button is pressed
pointermove	pointer changes coordinates, or when pressure, tilt, or button changes fire no other event

pointerup	active buttons state is left: i.e. stylus or finger leaves the screen, or mouse button released
pointercancel	pointer is determined to have ended, e.g. in case of orientation change, accidental input e.g. palm rejection, or too many pointers
pointerout	pointer moves out of an element (leaves its hit test boundaries). Also fired after pointerup event for no-hover supported devices, and afterpointercancel event
pointerleave	pointer moves out of an element and its descendants
gotpointercapture	when an element becomes target of pointer
lostpointercapture	when element loses pointer capture

Mouse events, pointer events, and touch events equivalence

For comparison, the following table shows the corresponding events from each of these input related APIs

Mouse event	*Touch event*	*Pointer event*
mousedown	touchstart	**pointerdown**
mouseenter		**pointerenter**
mouseleave		**pointerleave**
mousemove	touchmove	**pointermove**
mouseout		**pointerout**
mouseover		**pointerover**
mouseup	touchend	**pointerup**

Polyfill Library

Polyfill Library

It refers to a JavaScript **library** that implements an **HTML5** web standard, either an established standard (supported by some browsers) on older browsers, or a proposed standard (not supported by any browsers) on existing browsers.

Pointer Code Example

Pointer Code Example

```
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>PEP (Pointer Events Polyfill)</title>
  <meta name="viewport" content="width=device-width">
  <!-- include PEP -->
  <script src="https://code.jquery.com/pep/0.4.1/pep.js"></script>
</head>
<body>
<button id="b" touch-action="none">Test button!</button>
<p><output id="o"></output></p>
<script>
document.getElementById( "b" ).addEventListener( "pointerdown", function( e ) {
  document.getElementById( "o" ).innerHTML = "that was a " +
    e.pointerType + " " + e.type + " on a "+ e.target.nodeName;
} );
</script>
</body>
</html>
```

Pointer Code Example 2

Touch Handler Guide Lines

Touch Handler Guide Lines

It is good practice To use as few handlers as necessary in your application and keep the areas with touch handlers as tightly constrained as you can. That way the default scrolling mechanisms can take over as much as possible.

Input Introduction

Input Introduction

- Some users can easily type longer text on mobile devices.
- But it is still very touch to just type with thumbs.

- So we must make of Input entry easy for all the users.
- Also while entering data it should be clearly visible. It is much difficult to enter the small font data.

Keyboard Input

Phones and tablets do not have the benefit most desktops and laptops do, which is a large input device called the "keyboard". Virtual keyboards offered by mobile devices are physically smaller and usually more tedious to operate, but the great news is that HTML5 has already standardized a handful of input types which make them easier to use.

Input Types in HTML5

search	email	url	tel
number	range	date	month
week	time	datetime	datetime-local
color			

Text

The most common input type

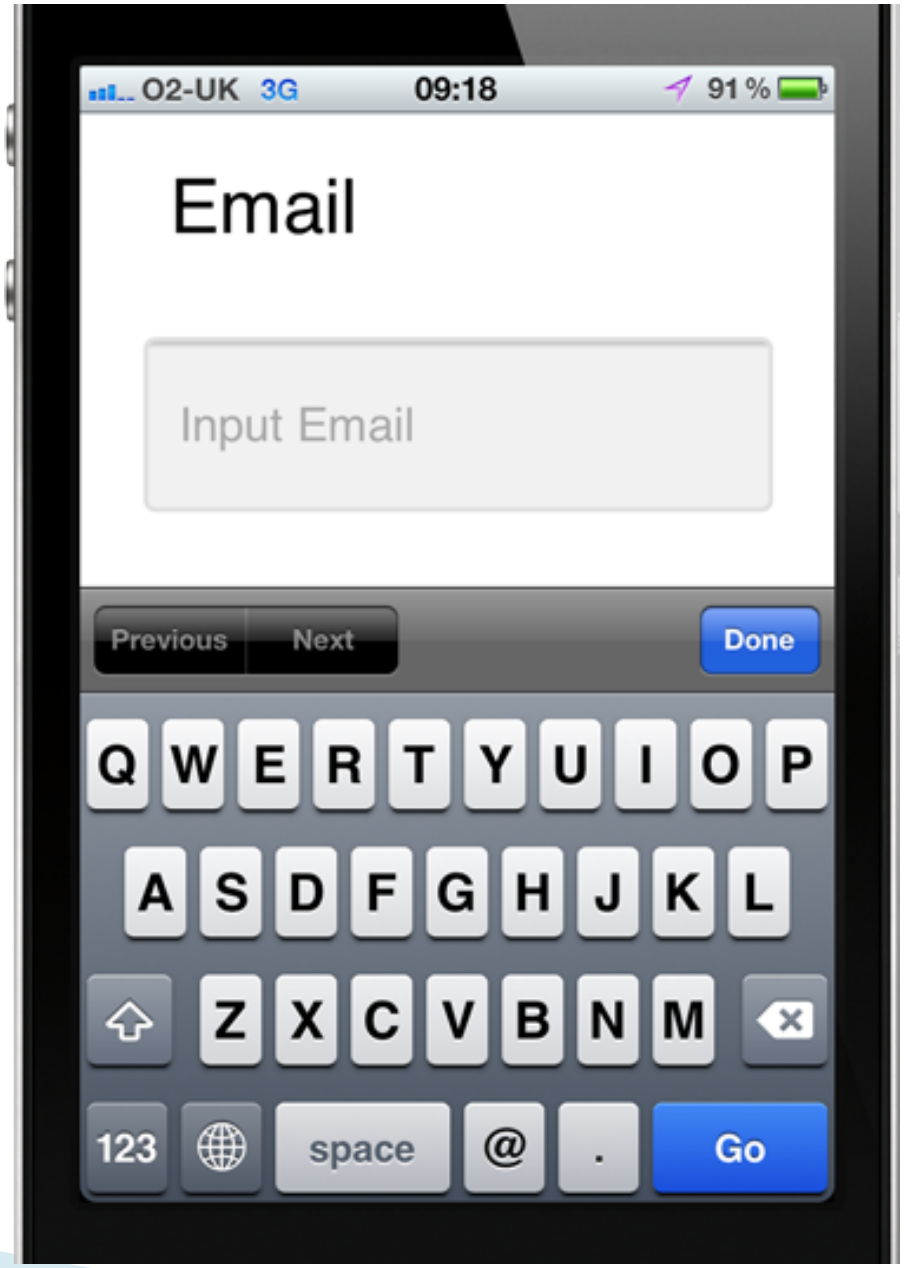
<input type="text">


O2-UK
17:19
99%
Text
Input Text
Previous
Next
Done
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
123
space
Go

Email

Entering an email on the standard keyboard is very difficult. When we use Input Type as email. We get a @ sign as well as the .com and other handy buttons to just make entering an email address as easy as possible.

<input type="email">


O2-UK 3G
09:18
91 %
Email
Input Email
Previous
Next
Done
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
123
space
@
.
Go

Tel

Tel input provides the user with ability to add the numbers , it provides ability to quickly add *, # , + signs as well.

```
<input type="tel">
}
```


O2-UK
17:23
99%
Tel
Input Tel
Previous
Next
Done
1
2 ABC
3 DEF
4 GHI
5 JKL
6 MNO
7 PQRS
8 TUV
9 WXYZ
+ * #
0

Number

Number input provides a key board to quickly add the numbers.

<input type="number">


O2-UK
17:24
99 %
Number
Input Number
Previous
Next
Done
1 2 3 4 5 6 7 8 9 0
– / : ; ( ) £ & @ ”
#+=
. , ? ! ’
ABC
space
Go

Date

Date input types makes it very easy to enter the date on Webpage. Using Calendar can be little difficult for users

<input type="date">


O2-UK
17:28
99 %
Date
2 Sep 2012
Previous
Next
Clear
Done
31 July 2010
1 August 2011
2 September 2012
3 October 2013
4 November 2014

Date & Time

Entering Date and Time is very Difficult but for mobile devices, using Date & Time input it becomes very easy to enter data.

<input type="datetime">


O2-UK
17:31
98 %
Date & Time
2 Sep 2012 17:31
Previous
Next
Clear
Done
Fri 31 Aug
Sat 1 Sep
Today
Mon 3 Sep
Tue 4 Sep
15
16
17
18
19
29
30
31
32
33

- Month
- Search

Regular Expression Validation

For some types like e-mail, there's some basic validation. But this is not very smart validation.

If you want to do better client side validation though,HTML-5 supports that too.

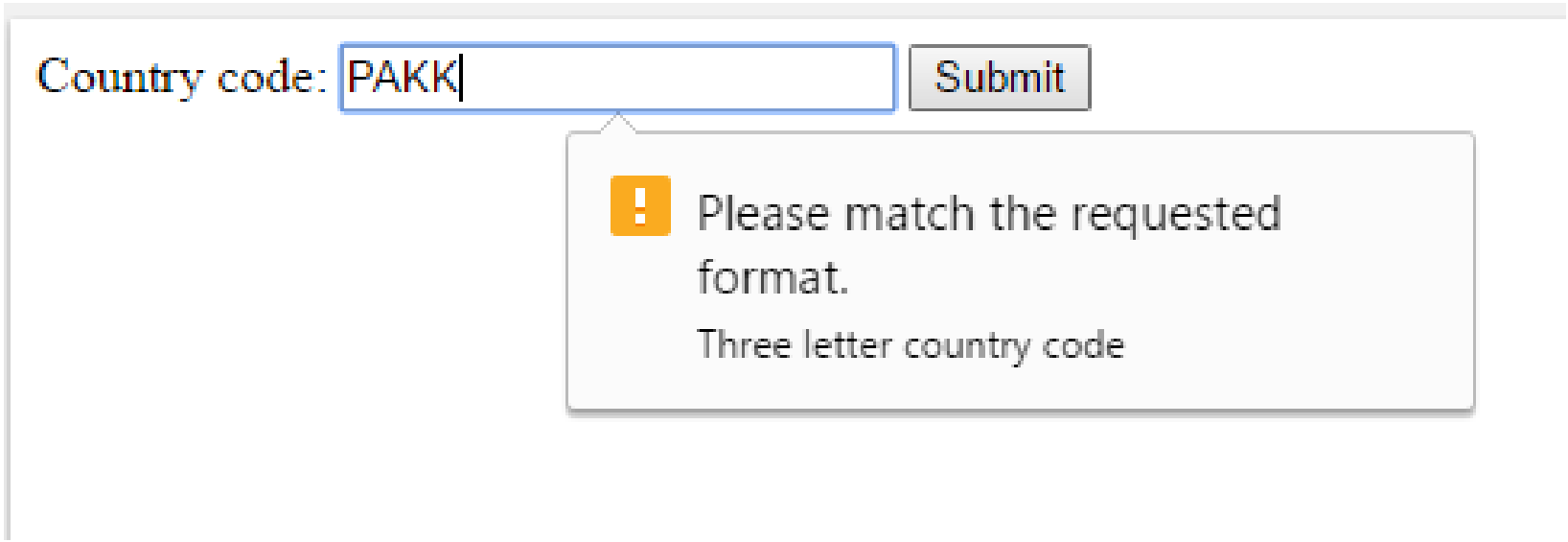
There's also a pattern attribute on the input element, that you can use to validate on the client side.

The pattern element takes a regular expression, as expressed by the java script regex syntax.

Example

For example, let's1presume we want to only allow Country code in a specific field in our mobile web application like PAK

```
<form action="demo_form.asp">
  Country code: <input type="text" name="country_code" pattern="[A-Za-z]{3}"
title="Three letter country code">
  <input type="submit">
</form>
```

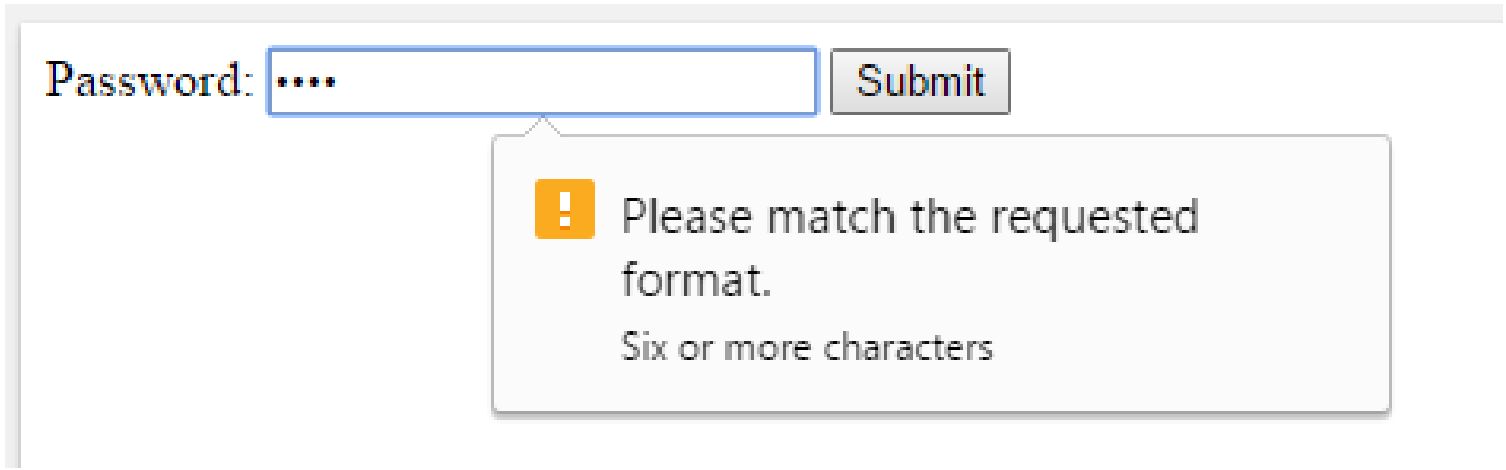

Country code:
PAKK
Submit
Please match the requested format.
Three letter country code

I can put in a regular expression for this, and now when I try to input a Country code of more then three letters and then submit, it pops up an error message that says I have to match the requested format.

Example 2

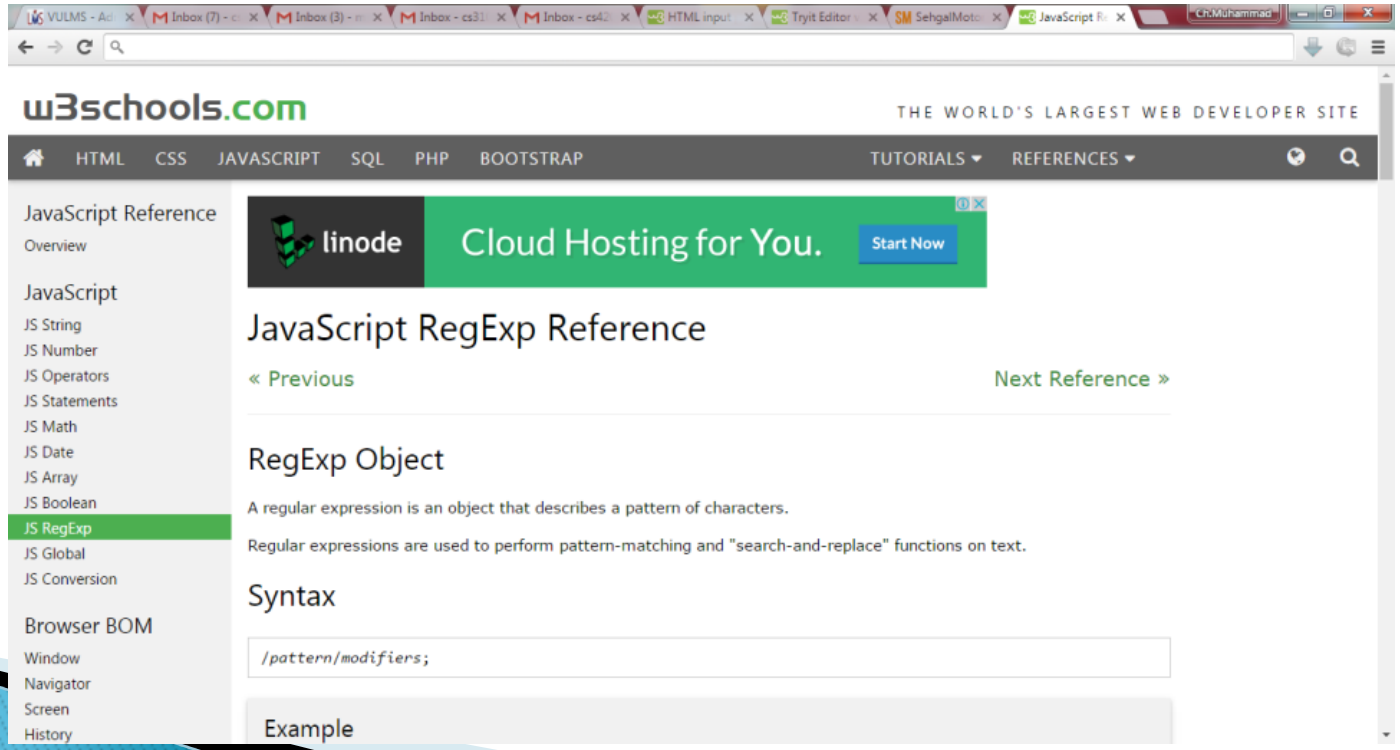
An <input> element with type="password" that must contain 6 or more characters:

```
<form action="demo_form.asp">
  Password: <input type="password" name="pw" pattern=".{6,}" title="Six or more
characters">
  <input type="submit">
</form>
```


Password:
••••
Submit
Please match the requested format.
Six or more characters

To Learn more about Regular Expressions

- http://www.w3schools.com/jsref/jsref_obj_regexp.asp



Two Important Features of Input

Two Important Features of Input

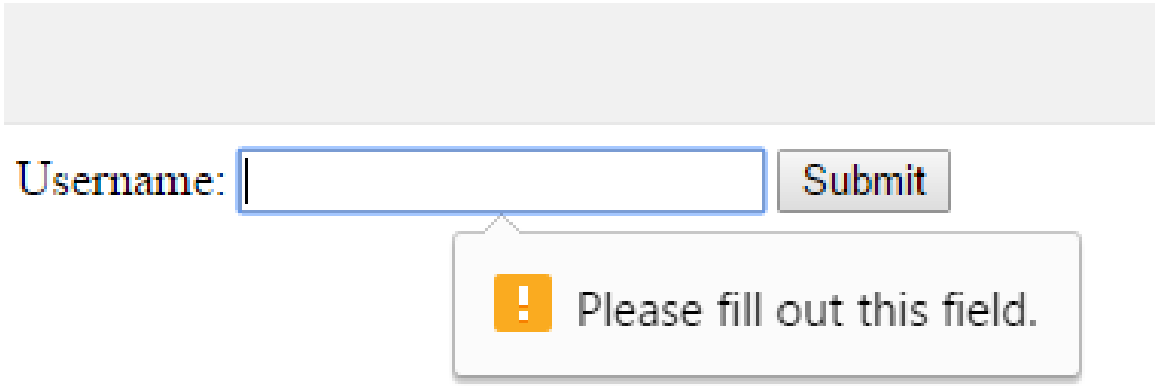
- There are two more features I wanted to mention on inputs.
- The required attribute and the place holder attribute.
- Required just lets me say, this field is required before you can submit the form. So,

Example

For example, if I want to make sure that a user enters the username,

```
<form action="demo_form.asp">
  Username: <input type="text" name="usrname" required>
  <input type="submit">
</form>
```

Now if user will try to submit this form with out entering the username he will get the error message



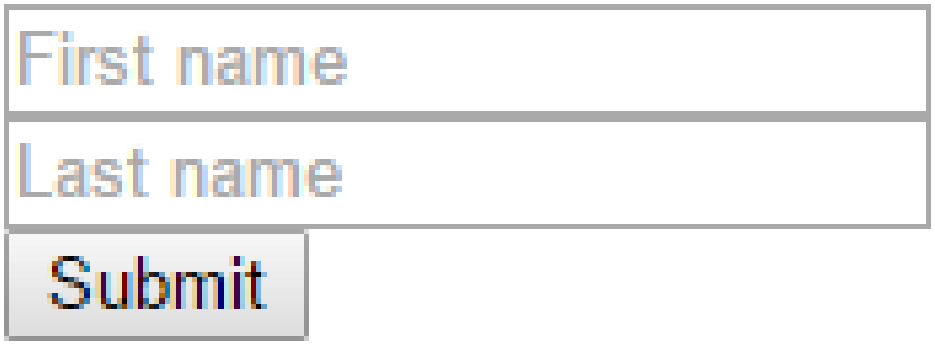
Placeholder Text

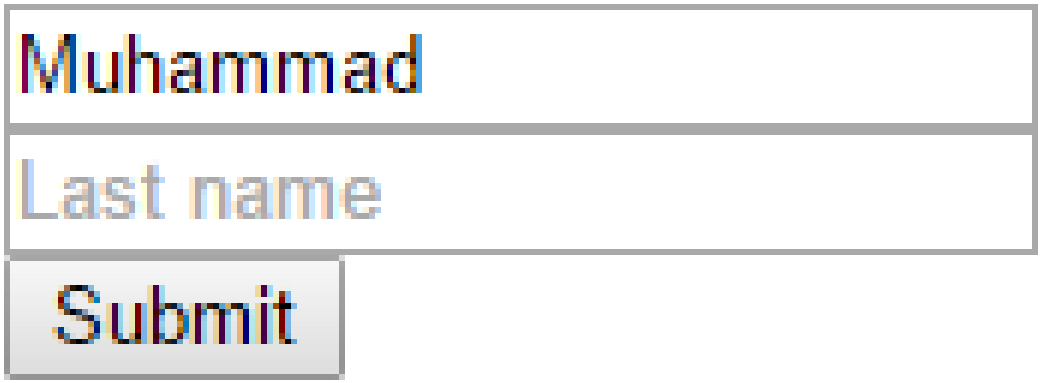
The other feature, is the placeholder text. You may want to give hints to your users. In some fields, they go away when they start typing. You can do that pretty easily by just setting a placeholder attribute.

- The placeholder attribute specifies a short hint that describes the expected value of an input field (e.g. a sample value or a short description of the expected format).
- The short hint is displayed in the input field before the user enters a value.
- **Note:** The placeholder attribute works with the following input types: text, search, url, tel, email, and password.

Example

```
<form action="demo_form.asp">
  <input type="text" name="fname" placeholder="First name"><br>
  <input type="text" name="lname" placeholder="Last name"><br>
  <input type="submit" value="Submit">
</form>
```


First name
Last name
Submit

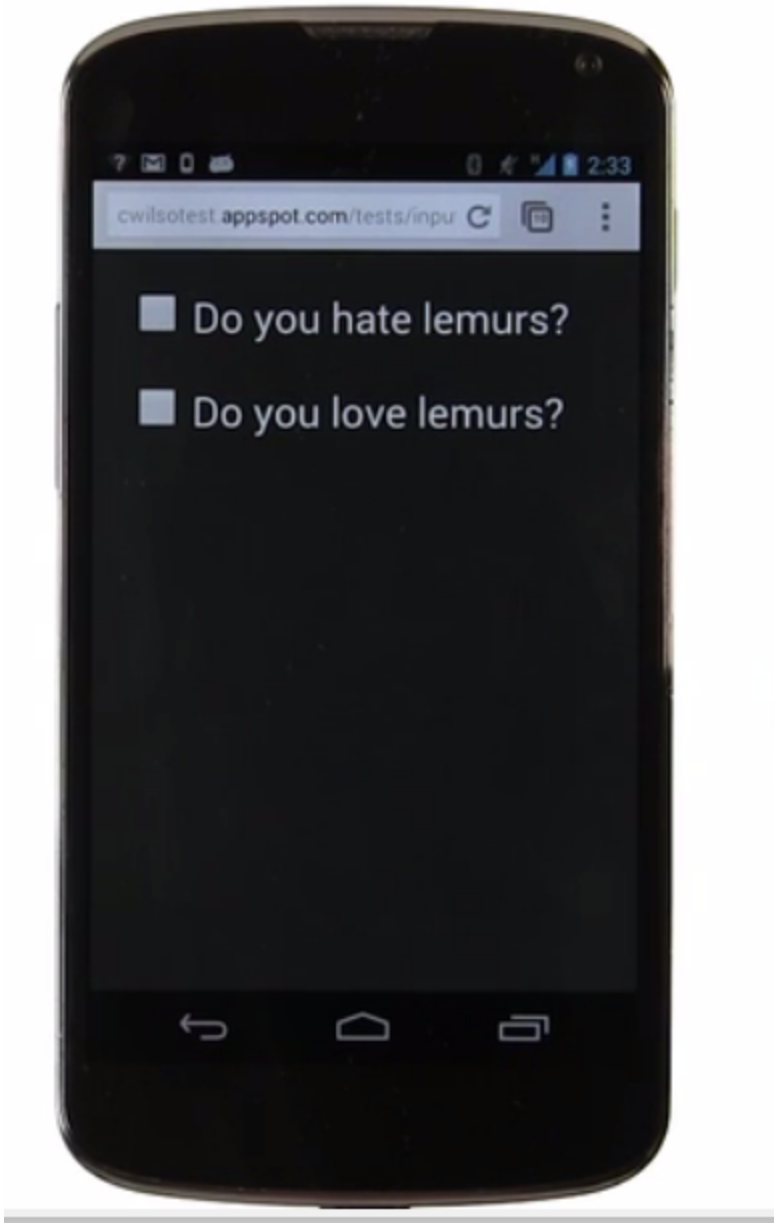

Muhammad
Last name
Submit

Input in older browsers

- The HTML spec says that unrecognized input types are to be
- treated as text. So, new input types will still work even if they are not supported on a particular browser.
- They'll just function like text inputs. Not quite as useful, but still functional.

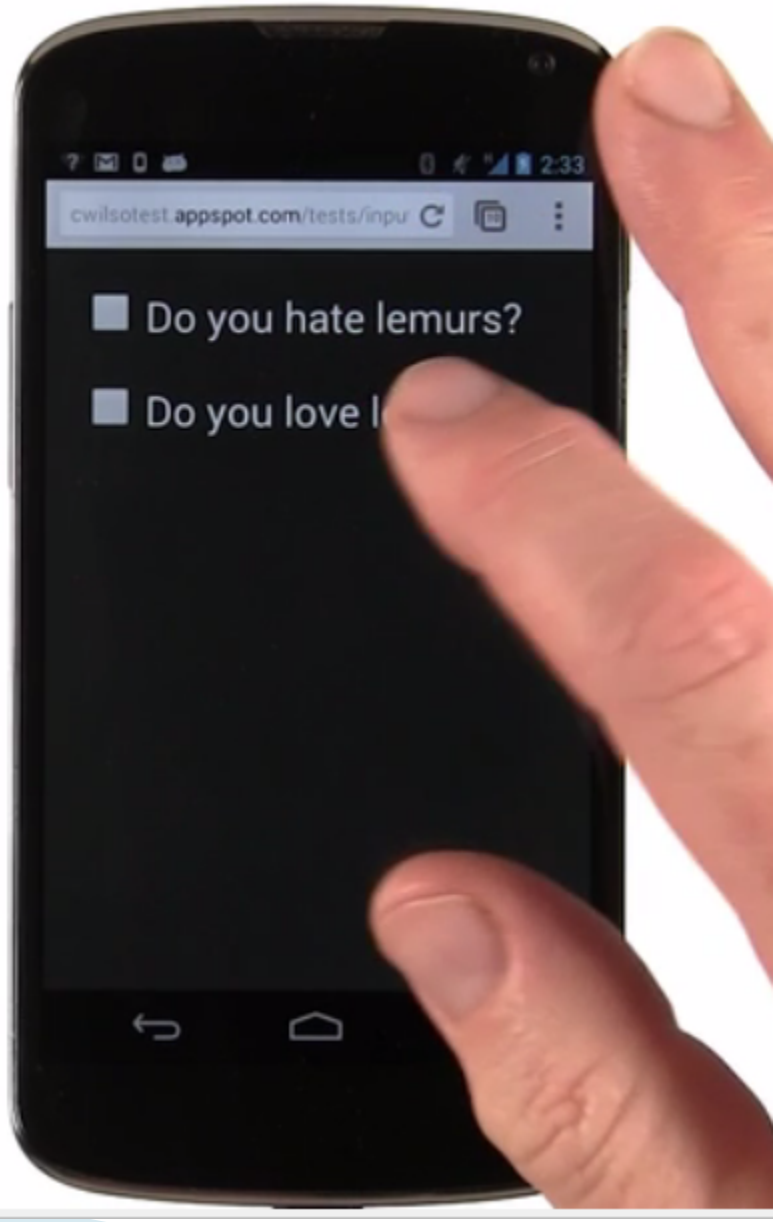
Wrapping in Label

You should always wrap label elements around your input elements, and their associated labeling texts, because it increases the touchable area of the control.


2:33
cwilsotest.appspot.com/tests/inpu
Do you hate lemurs?
Do you love lemurs?

- The only difference between these two controls is that one of them has a label element around it.

- But you'll notice, it's much easier to touch the one with the label. The one without a label, you have to actually touch the check box itself, in order to activate the control.


2:33
cwilsotest.appspot.com/tests/inpu
Do you hate lemurs?
Do you love l


<!DOCTYPE html>
<html>
<head>…</head>
<body>

<input type="checkbox">
" Do you hate lemurs?"

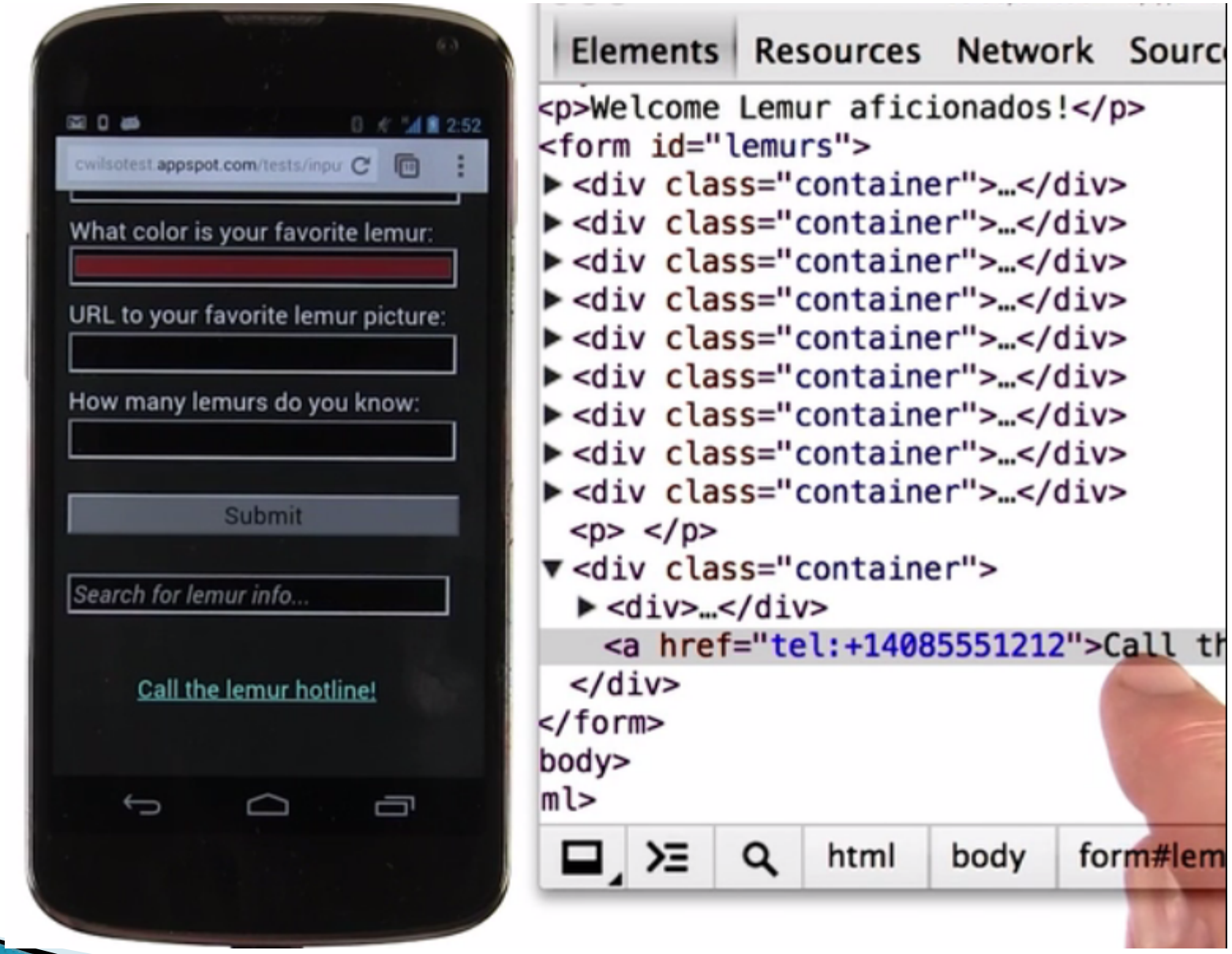
<label>
<input type="checkbox">
" Do you love lemurs?"
</label>

</body>
</html>

telecolon URL

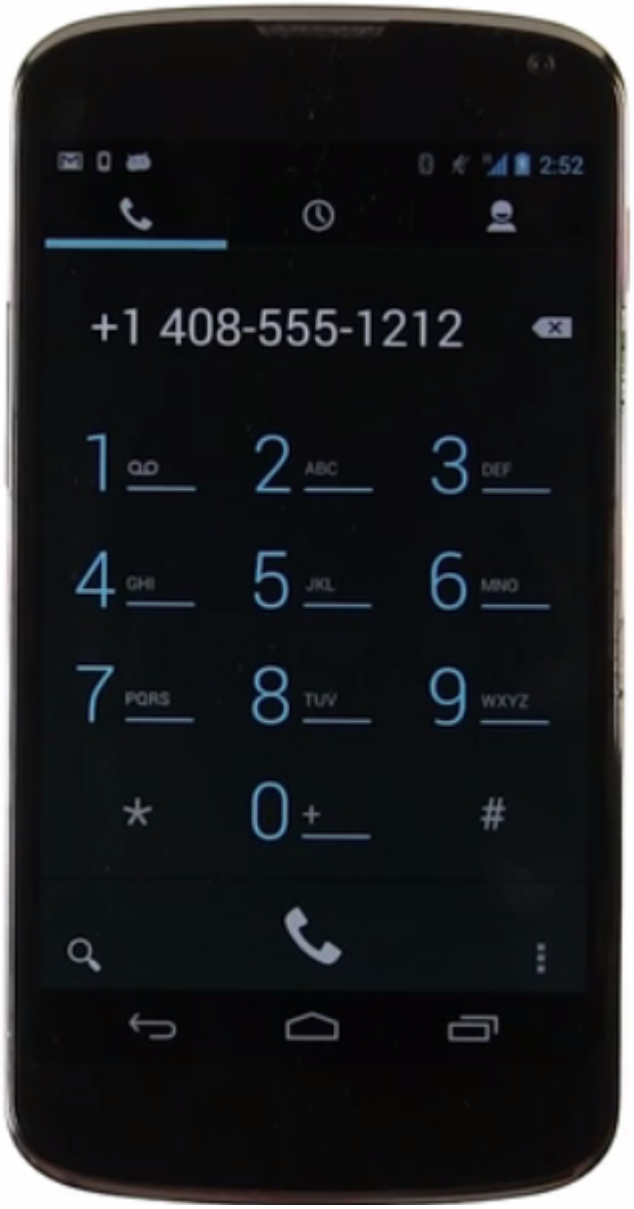
On mobile, you can use a telecolon URL to create a URL that when activated will actually dial a telephone number. This is a really handy feature, particularly for business websites. Customers can instantly dial you.

- Of course, on desktop, these URLs will typically fail, so you probably want to deactivate them.


2:52
cwilsotest.appspot.com/tests/inpu
What color is your favorite lemur:
URL to your favorite lemur picture:
How many lemurs do you know:
Submit
Search for lemur info...
Call the lemur hotline!
Elements
Resources
Network
Welcome Lemur aficionados!
<form id="lemurs">
…

…
<a href="tel:+14085551212">Call th

</form>
body>
ml>
html
body
form#lem


2:52
+1 408-555-1212
1
2 ABC
3 DEF
4 GHI
5 JKL
6 MNO
7 PQRS
8 TUV
9 WXYZ
*
0 +
#

Sensors on mobile

One of the most amazing things about the mobile platform is the incredible array of sensors these devices typically have.

This phone has a camera that can take stills and video, has audio input and output, it has a GPS, a compass, a tilt sensing accelerometer.

A Mobile device has far more sensors than a desktop or laptop typically has.

And all this, in a device that one can typically hold in the palm of his/her hand.

This enables the mobile platform to play host to incredibly engaging and interactive user experiences.

Camera Access

The camera is not just for taking pictures and making videos.

The camera can be a great way to get quick **input** into the mobile device.

For example you can snap a new profile picture using your mobile camera and upload it

- You capture a QR code to transfer a hyperlink

- You can capture a map or a business card.

Capture Extension

There is a simple way to get access to the camera and the audio input, the capture extension on the accept attribute on the file input element in HTML.

This will give you a button that opens up the system camera or Audio recorder app on a mobile device.

Accept Attribute

The accept attribute takes a comma-separated list of unique content types of files that are acceptable values for the file input type only. The comma separated values can include file extensions, MIME types without extensions,

```
<input type="file" accept="image/*">
```

(images only)

For Mobile Devices

```
<input type="file"
accept="image/*;capture=camera">
```

```
<input type="file"
accept="video/*;capture=camcorder">
```

```
<input type="file"
accept="audio/*;capture=microphone">
```

Drawbacks of Capture Extension

- The first problem is that this feature only works in mobile, not in desktop. On desktop, you only get the normal file open dialogue. Not access to the webcam.

- Secondly, the interface is a file open button, which is hard to style and make look the way you might want it to.

- Third issue is , taking you to a different application to take a picture or record some audio, makes it kind of hard to keep a consistent flow in your application.

Live Input

We can get Live Input, Audio and Video, directly under our App by using an API called getUserMedia.

Link:-
https://developer.mozilla.org/en/docs/Web/API/Navigator/getUserMedia

Example

```
var constraints = {video: true};

// prefix monkeypatching
navigator.getUserMedia = navigator.getUserMedia ||
  navigator.webkitGetUserMedia;

function successCallback(stream) {
  var video = document.querySelector("video");
  video.src = window.URL.createObjectURL(stream);
}

function errorCallback(error) {
  console.log("getUserMedia error: ", error);
}

navigator.getUserMedia(constraints, successCallback,
  errorCallback);
```

This is a simple example where we just call getUserMedia, passing some constraints that say we want video, and then we assign the resulting stream to a video element in the page. Of course, this just gives us a rectangle with live Video Input.

```
function snapshot() {
  var video = document.querySelector('video');
  var canvas = document.querySelector('canvas');

  // ensure the canvas matches the video dimensions
  canvas.width = video.videoWidth;
  canvas.height = video.videoHeight;

  // grab the video and draw it to the canvas
  var ctx = canvas.getContext('2d');
  ctx.drawImage(video, 0, 0);
}
```

We probably want to do something with it. If we want to grab Snapshots like if we want to implement a Camera Application, we can do this with a canvas context and DrawImage. Whenever we want to take a Snapshot, we simply grab the image from the video and we draw it to the camera with DrawImage.

```
function snapshot() {
  // ensure the canvas and img match the video
  canvas.width = video.videoWidth;
  canvas.height = video.videoHeight;
  img.height = video.videoHeight;
  img.width = video.videoWidth;

  // grab the video image and draw to the canvas
  var ctx = canvas.getContext('2d');
  ctx.drawImage(video, 0, 0);

  // encode the canvas contents to a data url.
  img.src = canvas.toDataURL('image/png');
}
```

Of course a canvas in different than an image. You can't directly copy a canvas and paste it somewhere else, or download it your hard drive or post it to your social networks. Or any of the other typical User flows centered around images, but it turns out, canvas has a1 handy function to encode itself as an image. The canvas toDataURL method.

Here, instead of displaying the canvas itself, we're copying the canvas contents as a data earl to the source of an image tag. And if you want to save these images to the Mobile's local storage.

Navigator.getUserMedia()

Syntax

```
navigator.getUserMedia(constraints,
successCallback, errorCallback);
```

Parameters

Constraints

A MediaStreamConstaints object specifying the types of media to request, along with any requirements for each type.

- To *require* a capability, use the keywords min, max, or exact (a.k.a. min == max). The following demands a minimum resolution of 1280x720

```
{
  audio: true,
  video: {
    width: { min: 1280 },
    height: { min: 720 }
  }
}
```

successCallback

When the call succeeds, the function specified in the successCallback is invoked with theMediaStream object that contains the media stream. You may assign that object to the appropriate element and work with it, as shown in the following example:

```
function(stream) {
    var video = document.querySelector('video');
    video.src = window.URL.createObjectURL(stream);
    video.onloadedmetadata = function(e) {
        // Do something with the video here.
    };
}
```

errorCallback

When the call fails, the function specified in the errorCallback is invokedwith a MediaStreamError object as its sole argument

Error	Description
PermissionDeniedError	Permission to use a media device was denied by the user or the system.
NotFoundError	No media tracks of the type specified were found that satisfy the constraints specified.