Touch Forward

Developing Awesome Cross-Browser Touch Interactions

Bill Fisher

@fisherwebdev

#touchfwd

Super F*cking Important

- yeah, it's important.

http://commons.wikimedia.org/wiki/File:071228_human_hands.JPG

http://commons.wikimedia.org/wiki/File:Touchscreen_IMG_2796.jpg

The Agenda

- click vs. mouseend vs. touchend
- touch events overview and how-to
- animations with hardware accelerated CSS and/or requestAnimationFrame
- plugins & libraries for the lazy & the sane
- pointer events and our beloved Internet Explorer
- haptic feedback
- the post-touch world to come

The Agenda

- click vs. mouseend vs. touchend
- touch events overview and how-to
- animations with hardware accelerated CSS and/or requestAnimationFrame
- plugins & libraries for the lazy & the sane
- pointer events and our beloved Internet Explorer
- haptic feedback
- the post-touch world to come

standard order of events in a tap

- touchstart
- touchmove (+n)
- touchend
- mouseover
- mousemove (usually once)
- mousedown
- mouseup
- click
- mouseout

delays after touch events

- Mobile Safari: ~380ms
- Android Browser: ~300ms
- Chrome on Android: ~240ms
- Opera Mobile on Android: ~260ms
- FF on Android: ~310ms

it's not okay.

- audio: 10ms distracting, 20ms echo
- video: 33ms frame rate, 17ms screen refresh
- HCI research: 100ms < not "instantaneous"

response times

- 100ms: the threshold of the feeling that the app is responding instantaneously
- 100ms - 1000ms: user loses the feeling of operating directly on the data
- 1000ms: user's flow of thought is interrupted

http://www.useit.com/papers/responsetime.html

viewport meta tag

<meta name="viewport" content="width=device-width,
initial-scale=1.0, maximum-scale=1.0, user-scalable=0" />

eliminates delay in some browsers:

- Chrome for Android
- Firefox for Android

it would be great if this effect was more widespread!

a pretty clear choice

- Play it safe with click and create a laggy experience
- Do some work with detecting touch event support and create something that responds nicely

The Agenda

- click vs. mouseend vs. touchend
- touch events overview and how-to
- animations with hardware accelerated CSS and/or requestAnimationFrame
- plugins & libraries for the lazy & the sane
- pointer events and our beloved Internet Explorer
- haptic feedback
- the post-touch world to come

touch event support

- Mobile Safari
- Android Browser
- Opera Mobile
- Chrome for Android ⬅ also supports v.2
- Chrome for iOS
- Firefox for Android ⬅ also supports v.2
- Blackberry Browser
- *NOT: Opera Mini or Internet Explorer*

w3c touch events v.1: the events

- touchstart
- touchmove
- touchend
- touchcancel

w3c touch events v. 1: event properties

- inherits from UIEvent (...so shiftKey, etc.)
- touches (Array)
- targetTouches (Array)
- changedTouches (Array)

w3c touch events v. 1: touch properties

- pageX / pageY
- clientX / clientY
- screenX / screenY
- target
- identifier

w3c touch events v. 1: methods

- preventDefault ← this is your friend
- stopPropagation & other UIEvent methods
- initTouchEvent

w3c touch events v.2: new events & properties

- new events: touchenter / touchleave
- new property: relatedTarget

w3c touch events v.2: new touch properties

- radius X / radiusY
- rotationAngle
- force

(vendor prefixed in Firefox and Chrome)

ok, now what?

We can get the data from the events.

But what do we do with it?

setting up the page: html and javascript

```
<meta name="viewport" content="width=device-width,
initial-scale=1.0, maximum-scale=1.0, user-scalable=0"/>
```

```
/* if we want to lock down the scrolling entirely */
window.addEventListener("touchmove", function(e){
  e.preventDefault();
});
```

setting up the page: css

```
html {
  -webkit-user-select: none;
  -moz-user-select: none;
  -ms-user-select: none;
  -webkit-touch-callout: none;
  -webkit-user-drag: none;
  -webkit-tap-highlight-color: rgba(0,0,0,0);
}
```

The Agenda

- click vs. mouseend vs. touchend
- touch events overview and how-to
- animations with hardware accelerated CSS and/or requestAnimationFrame
- plugins & libraries for the lazy & the sane
- pointer events and our beloved Internet Explorer
- haptic feedback
- the post-touch world to come

Hey U get on the GPU

- hardware accleration
- multicore programming
- much improved performance

CSS3 hacks for hardware acceleration

- any transition
- any 3D transform
- translate3d(0,0,0) or translateZ(0)

testing for hardware acceleration

- Chrome about:flags: Composited render layer borders & FPS counter
- Start Safari in Terminal: `CA_COLOR_OPAQUE=1 /Applications/Safari.app/Contents/MacOS/Safari`
- Start iPhone Simulator in Terminal: `CA_COLOR_OPAQUE=1 /Developer/Platforms/iPhoneSimulator.platform/Developer/Applications/iPhone\ Simulator.app/Contents/MacOS/iPhone\ Simulator`

event handlers

- throttled frame rates: with a maximum rate or ideally with requestAnimationFrame
- temporarily name anonymous functions so they may be profiled
- separate reading and modifying of DOM values

things to avoid

- DOM manipulation: hit the DOM once
- accidental browser layout reflow
- animated css shadows and gradients

gesture events

- only available in Mobile Safari
- two fingers
- rotation, scale
- convenient for prototyping, but not worth it for production

...but we want gestures

- difficult to determine user intention in a touch environment
- lots of logic and a bit of math to maintain
- libraries can help

The Agenda

- click vs. mouseend vs. touchend
- touch events overview and how-to
- animations with hardware accelerated CSS and/or requestAnimationFrame
- plugins & libraries for the lazy & the sane
- pointer events and our beloved Internet Explorer
- haptic feedback
- the post-touch world to come

Plugins and Libraries

- Touchy
- Hammer.js
- jGestures
- Zepto
- https://github.com/bebraw/jswiki/wiki/Touch

Touchy

- a highly configurable jQuery plugin for managing touch events
- sponsored by Hot Studio
- vanilla JS version, mouse fallback, and IE10 support coming soon

Touchy

- loosely based on native iOS gesture recognizers: drag, swipe, pinch, rotate, longpress
- event handlers receive the "phase" of the event in a parameter, in addition to other data
- touchyjs.org ⬅ check it out!
- @touchyjs

Hammer.js

- http://eightmedia.github.com/hammer.js/
- Very similar to Touchy
- Mouse support, including IE8 & IE9
- Covers a few more events (tap, double tap)
- Less flexible for custom interactions
- jQuery plugin or vanilla JS

The Agenda

- click vs. mouseend vs. touchend
- touch events overview and how-to
- animations with hardware accelerated CSS and/or requestAnimationFrame
- plugins & libraries for the lazy & the sane
- pointer events and our beloved Internet Explorer
- haptic feedback
- the post-touch world to come

W3C Pointer Events

- w3c proposal from Microsoft
- abstracts mouse, touch and stylus into unified event model
- inherits from UIEvent
- independent events for each touch
- implemented in IE10 for Windows 8

stylin' with a stylus

Pointer Events improve stylus interactions:

- pressure
- tiltX / tiltY
- allows expressiveness previously not attainable

gaming consoles

Pointer Events could improve
browsing with a game controller

- support for up to five buttons on each pointer
- two users with two controllers could be interacting with the page independently

Pointer Events: no gestures?

- How do we create gestures that require the association of multiple pointers?
- Can two people both do multitouch?

MSPointerEvent & MSGestureEvent in IE10

- standard gestures: long press, ...
- a lot of boilerplate for custom gestures
- minimum of two fingers
- event data values are matrix values
- inertia is available by default!

MSGestureEvent boilerplate: css

```
body {
  /* prevent default gestures */
  -ms-touch-action: none;

  /* prevent text selection */
  -ms-user-select: none;
}
```

MSGestureEvent boilerplate: javascript

```
var customGesture = new MSGesture();
var elem = document.getElementById("thang");

// assign the element to the gesture object
customGesture.target = elem;

elem.addEventListener("MSPointerDown", function (ev) {
  // each pointer must be added to the gesture
  customGesture.addPointer(ev.pointerId);
});
```

MSGestureEvent event handler

```
elem.addEventListener("MSGestureChange", function (ev) {
  // get the current css transform matrix
  var matrix = new MSCSSMatrix(ev.target.style.transform);

  // apply a new matrix built from the old
  ev.target.style.transform = matrix
    .rotate(ev.rotation * 180 / Math.PI)
    .scale(ev.scale)
    .translate(ev.translationX, ev.translationY);
});
```

pointer.js

- MSPointerEvents polyfill
- does not change the target of the event on touchmove / mousemove
- by Boris Smus (Google)

PointerEvents.js

- MSPointerEvents polyfill, but it does attempt to fire events on new elements during touchmove / mousemove.
- by Daniel Freedman (Google)

The Agenda

- click vs. mouseend vs. touchend
- touch events overview and how-to
- animations with hardware accelerated CSS and/or requestAnimationFrame
- plugins & libraries for the lazy & the sane
- pointer events and our beloved Internet Explorer
- haptic feedback
- the post-touch world to come

getting hip to haptics

```
var elem = document.querySelector(".vibe");

elem.addEventListener("touchstart", function () {
  elem.style.background = "#f00";
  if (navigator.vibrate) navigator.vibrate(12);
});

elem.addEventListener("touchend", function () {
  elem.style.background = "transparent";
});
```

post-touch

- touch is limited by the “pictures under glass” problem
- feelscreens could help
- motion-capture could rival touch, but will not supplant it without haptics
- pointer events might work in the near term with Kinect-like devices

FTW

- click is not good enough.
- there's all kinds of awesome in touch events and hardware accelerated css!
- IE10 is kind of cool, but pointer events are way cool.
- fight for the web by making it awesome!

Thanks!

@fisherwebdev