TIZEN™ DEVELOPER CONFERENCE
MAY 7–9, 2012

# Write a Touch-friendly HTML5 App

Hongbo Min, Intel
Junmin Zhu, Intel
Yongsheng Zhu, Intel

# Agenda

- Background
- Touch vs. Mouse
- Part I: UI Layout
- Part II: Event Handling
- Touch in HTML5 Framework
- Q/A

# Background

- Touching Experience
  - First introduced by Apple in iOS 2.0
  - Widely supported on mobile platforms today
  - Supported by both native app and web app

# Background

- Touch Event API (W3C Spec)
  - Define low-level touch events for web app
  - 4 touch events in Version 1 (CR)
    - **`touchstart,touchend,touchmove,touchcancel`**
  - 2 more touch events in Version 2 (Raw File)
    - **`touchenter,touchleave`**

- Gesture Event API (Apple)
  - **`gesturestart,gestureend,gesturechange`**

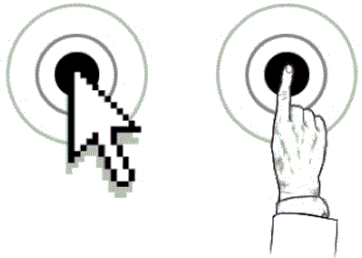
# Background

- Touch Event Handling
  - Follow DOM Level 2 event handling style
  - Add touch event listener on a DOM element

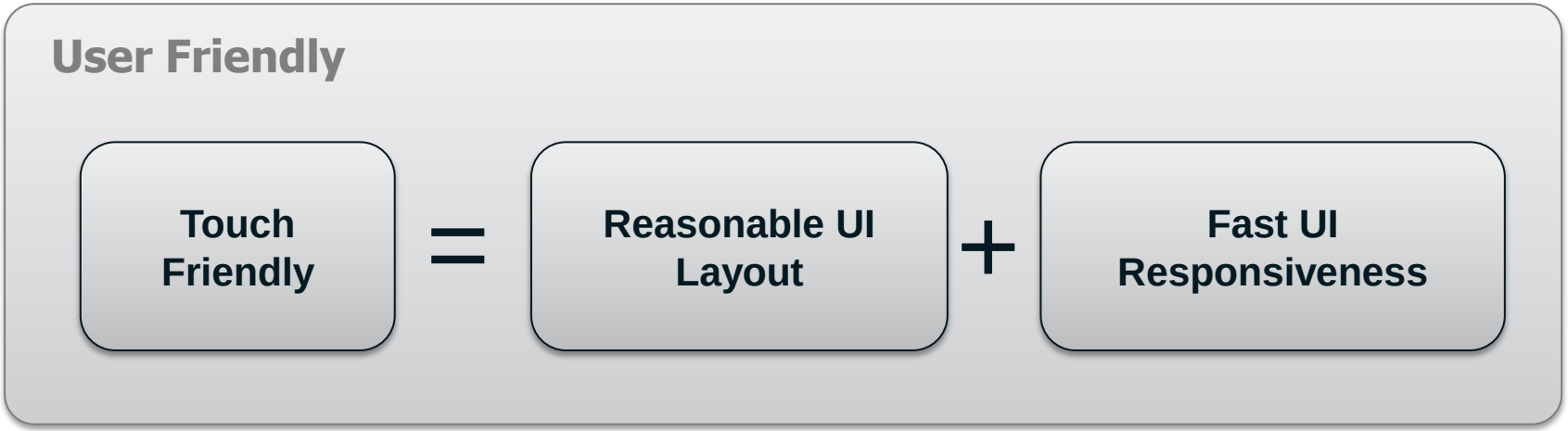
```
var elem = document.getElementById("canvas");
elem.addEventListener("touchmove", function(event){
    // handle touch move event
}, false);
```

# Touch vs. Mouse

- Touch-based interaction mode
  - No precise, fixed touch position on the screen
  - Require large space for selectable element
  - Support multiple touch positions
  - Support gestures like tap, pinch, swipe
  - No default focused element
  - No hover event for element (version 1)



# Touch Friendly App



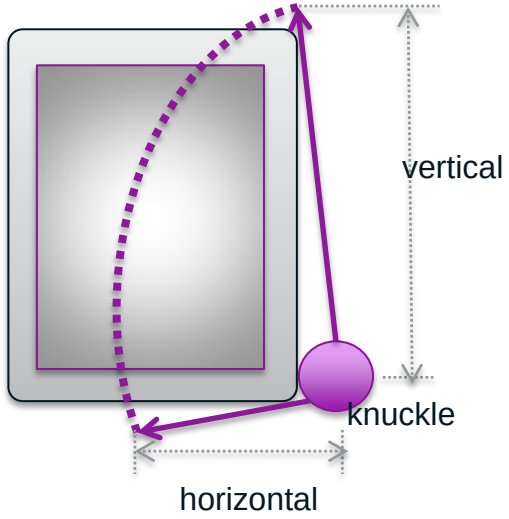
# Part I: UI Layout

- Make sure the clickable area is large enough
  - Make it easier for the user to select an item
  - For example, the optimal size of a button or a list item is ~**9x9** $mm^2$ on small touch screen
  - OS vendors have design guidelines for touch target size




# Part I: UI Layout

- Limit scrolling to one direction, vertical preferred
  - Keep content in one single column
  - Vertical scrolling is preferred on a small screen



# Part I: UI Layout

- Auto-hide browser's address bar

```
// Hides mobile browser's address bar when page is loaded.
window.addEventListener('load', function(e) {
    setTimeout(function() { window.scrollTo(0, 1); }, 1);
}, false);
```

# Part I: UI Layout

- Targeting screen size
  - The media tag describes the conditions for applying the CSS given in the href attribute

```
<link rel="stylesheet" media="handheld, only screen and
(max-device-width: 320px)" href="mobile.css">
```

  - The following targets screen width between 600px and 800px:

```
<link rel="stylesheet" media="only screen and (min-width:
600px) and (max-width: 800px)" href="ipad.css">
```

# Part I: UI Layout

- Fullscreen mode
  - Prefer to toggle Fullscreen for web apps, especially for HTML5 games
  - W3C fullscreen API is still not widely supported on mobile browser

# Part I: UI Layout

- Disable Zooming
  - Avoid triggering browser default behavior for zooming

```
<meta name="viewport" content="width=device-width,
 initial-scale=1.0, user-scalable=no">
```

# Part II: Event Handling

- Prevent Default Behavior
  - Avoid triggering browser itself behavior for the touch events
  - Send message preventDefault to event object
  - For example, to prevent scrolling by browser

```
// Call preventDefault on touchmove or touchstart
document.addEventListener("touchmove", function(event)
{ event.preventDefault(); }, false);
// Note the following will also prevent mouse events from firing
document.addEventListener("touchstart", function(event)
{ event.preventDefault(); }, false);
```

# Part II: Event Handling

- Multi Touch Events
  - A new touchstart/touchmove/touchend event is triggered each time a finger places down/moves/lifts up screen
  - Make use of touches, targetTouches and changedTouches lists
  - Can be recognized as gesture events

# Part II: Event Handling

- Handle multi-touch efficiently
  - Example: a canvas drawer to handle multi fingers drawing

```
canvas.addEventListener('touchmove',
function(event) { drawCanvas(event.touches); }, false);
```

```
var touches = [];
canvas.addEventListener('touchmove',
     function(event) { touches = event.touches; }, false);
function drawLoop() {
     requestAnimationFrame(drawLoop);
     drawCanvas(touches);
}
drawLoop();
```

1. Less CPU overhead
2. Better performance

# Part II: Event Handling

- Handle orientation changes
  - Different UI appearances for landscape mode and portrait mode
  - iOS can trigger onorientationchange event

```
<body onorientationchange="updateOrientation();">
```

  - Tizen can trigger deviceorientation event

```
window.addEventListener("deviceorientation", function(event){
    // process event.alpha, event.beta and event.gamma
}, true);
```

  - Android doesn't support onorientationchange, please resize event instead

# Touch in HTML5 Framework

- Touch Event abstraction
  - Hide the differences of event handling on different platforms
  - Unify event-handling model for touch and mouse events
  - No need to handle raw touch or mouse events

# Touch in HTML5 framework

- jQuery Mobile provides
  - High level touch events
    - **`tap,taphold,swipe,swipeleft,swiperight`**
  - Virtual mouse events to abstract away mouse and touch event
    - **`vmouseclick,vmousedown,vmouseup,...`**
  - Orientation change event support
  - No multi-touch gestures support

# Q/A

# Reference


[1] Building a Touch-Friendly HTML5 Site, Erik Klimczak
[2] The Touch Events, Peter-Paul Koch
[3] Multi Touch Development , Boris Smus
[4] Touch Events, Mozilla Developer Network
[5] Target Size Study for Small Touchscreen Devices, Pekka Parhi, etc
[6] Touch Target Size, Luke Wroblewski
[7] Mobile Web Best Practices 1.0, W3C Recommendation
[8] Mobile Web Application Best Practices, W3C Recommendation
[9] Safari Web Content Guide - Handle Event, iOS Developer Library
[10] Using Gesture Library, gestureworks.com
[11] Touch and Gesture on iPhone, Android and More, Colin Snover
[12] jQuery Mobile - Events, jqurymobile.com

TIZEN™ DEVELOPER CONFERENCE
MAY 7–9, 2012