

HTML5 Features on Tizen

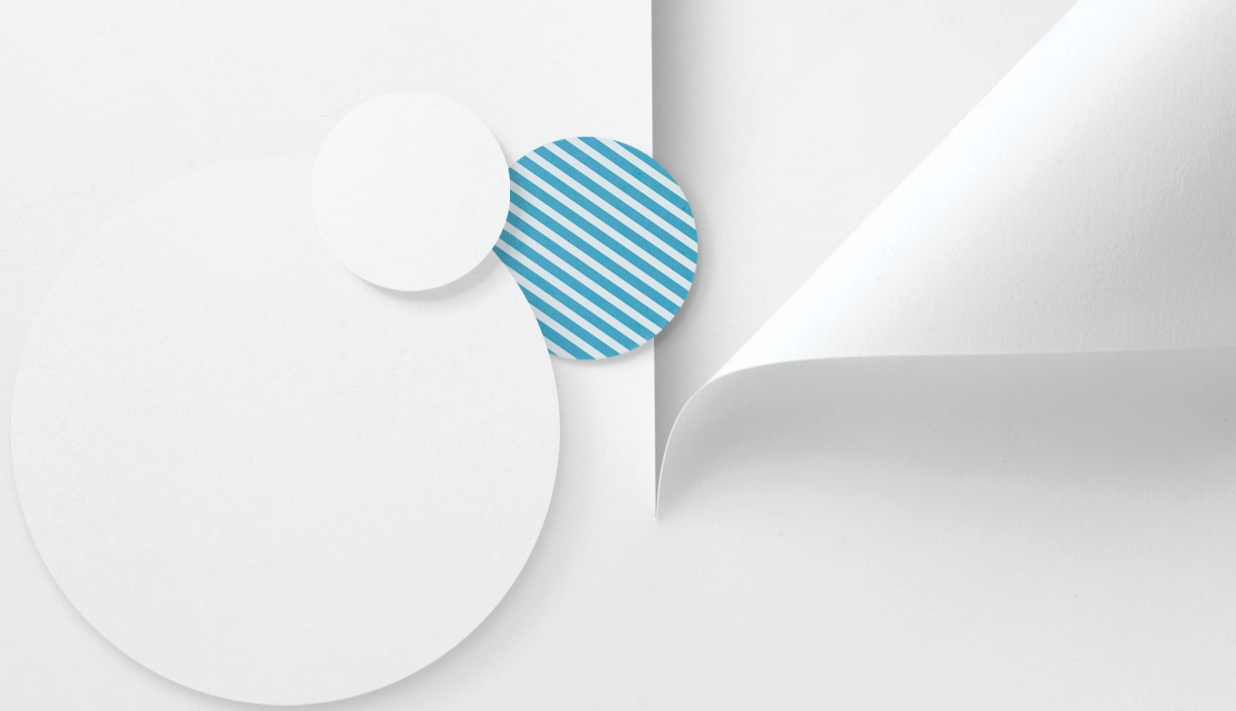
Wonsuk Lee, Samsung Electronics



TIZEN™
**DEVELOPER
CONFERENCE**
2013
SAN FRANCISCO

Table of Contents

- **Brief overview of HTML5**
- **Introducing Tizen web framework**
- **New HTML5 features on Tizen**
- **Upcoming cutting-edge HTML5 features**



Brief overview of HTML5

What is HTML5?

Structure and Semantic



APIs

<header>

<nav>

<section id="content">

<article>

<article>

<article>

<footer>



Goal of HTML5: Standards for web app♪

New Markups in HTML5

- **New Markup Elements – for better structure**

- `<article>`, `<aside>`, `<command>`, `<details>`, `<summary>`, `<figure>`, `<figcaption>`, `<footer>`, `<header>`, `<mark>`, `<meter>`, `<nav>`, `<progress>`, `<ruby>`, `<rt>`, `<rp>`, `<section>`, `<time>`

- **New Media Elements – for media content**

- `<audio>`, `<video>`, `<source>`, `<embed>`

- **The Canvas Element – for drawing**

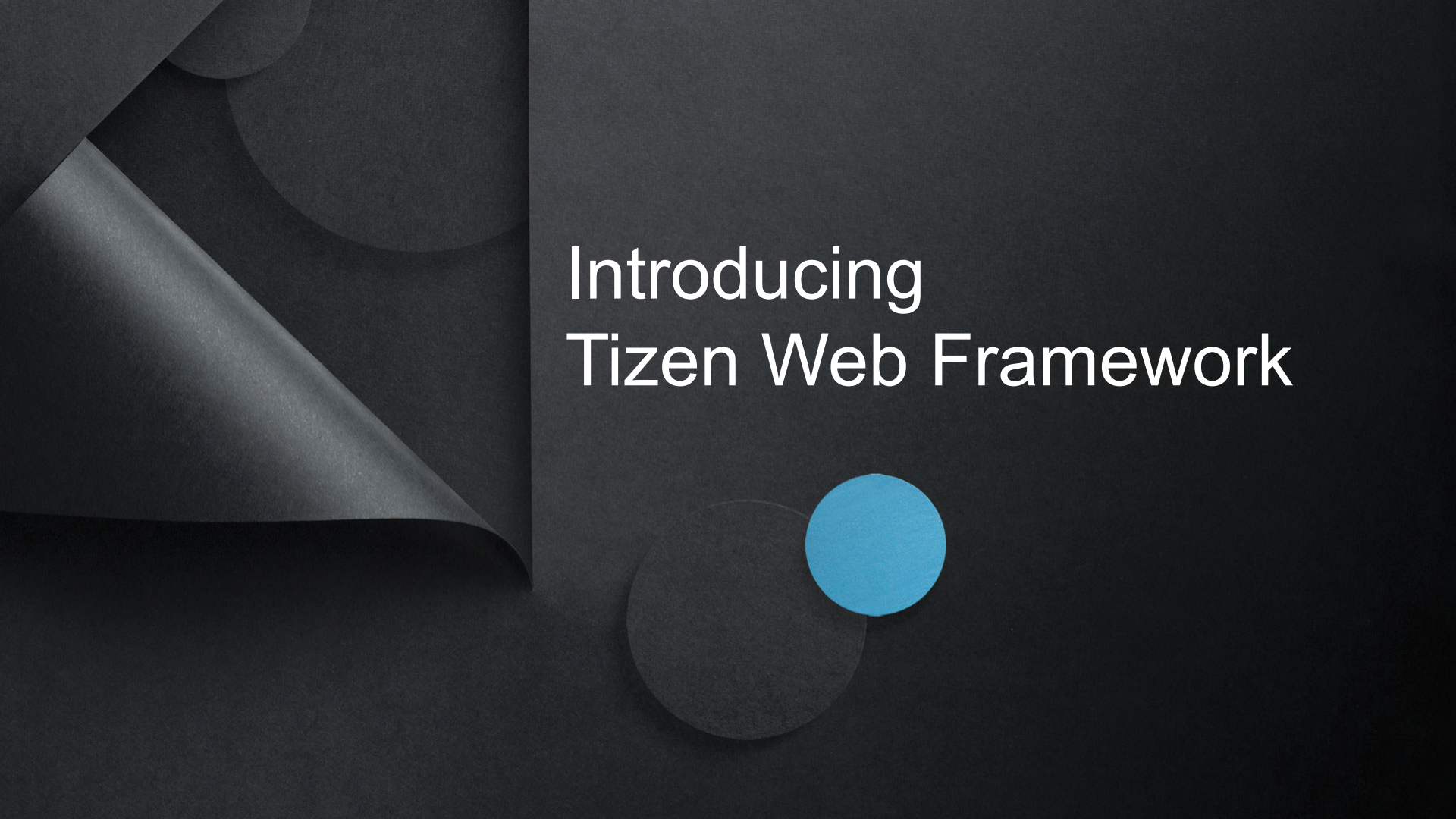
- `<canvas>`

- **New Form Elements and Input type attribute values**

- `<datalist>`, `<keygen>`, `<output>`
- New input type attribute values: **email**, **url**, **number**, **range**, **date**, **month**, **week**, **time**, **datetime**, **datetime-local**, **search**, **color**, etc

New APIs in HTML5

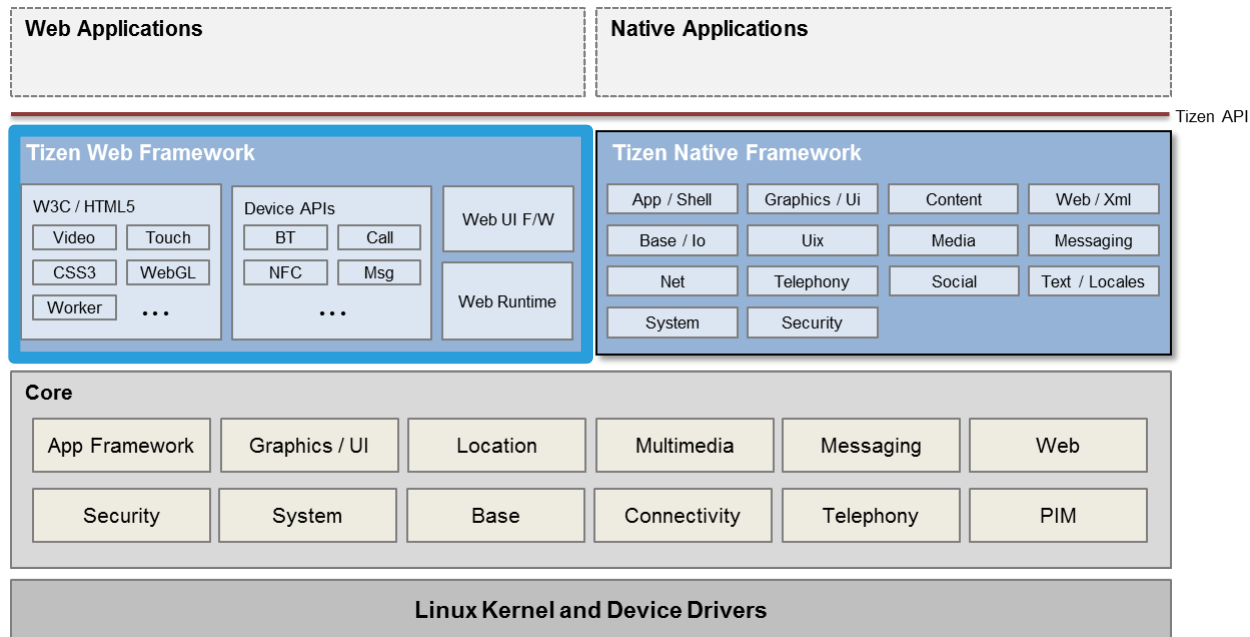
- **Graphics**
 - 2D Canvas API, Timing control for script-based animations API
- **Media**
 - HTML Media Capture, Web Real-Time Communication APIs, Web Audio API
- **User interactions**
 - Touch Event API, Vibration, Web Notifications
- **Data Storage**
 - Web Storage, File Reader API, File Writer API, File System API, Indexed Database API, **WebSQL Database**
- **Personal Information Management**
 - **Contacts API**, **Calendar API**
- **Sensors and hardware integration**
 - Geolocation API, DeviceOrientation Event, Battery Status Event, **System Information API**
- **Network**
 - Server-Sent Events, WebSocket API, Network Information API
- **Communication**
 - **Messaging API**, HTML5 Web Messaging, **Web intents**
- **Performance & Optimization**
 - Navigation Timing, Resource timing, Page Visibility API, Web Workers



Introducing Tizen Web Framework

Tizen Architecture

- Tizen is a Linux-based platform providing both Web and native APIs



Tizen Architecture

- **Web framework** provides state-of-the-art HTML5/W3C APIs, Web UI framework, Web Runtime and Tizen device APIs, such as Bluetooth and NFC
- **Native framework** supports full-featured native application development and provides a variety of features, such as background applications, IP push, and TTS
- **Tizen core** is an underlying layer for Web and native, and also for HW adaptation

Tizen Web Framework

- **W3C standard Web APIs** – W3C/HTML5 Markup, CSS, JavaScript APIs
- **Supplementary APIs** – De-facto APIs (Khronos, Mozilla, etc.)
- **Web Runtime** – Allowing Web applications to run outside the browser
- **Tizen Device APIs** – Advanced access to the device's platform capabilities
- **UI Framework** – Tools, such as widgets, events, effects, and animations

HTML5TEST.COM Score of Tizen

current

		Score	Bonus
BlackBerry 10 »	BlackBerry Q10 or Z10	485	11
Chrome 25 »	All Android 4 devices	417	11
Opera Mobile 12.10 »	Multiple platforms	406	12
Firefox Mobile 19 »	Multiple platforms	399	14
iOS 6.0 »	Apple iPhone, iPad and iPod Touch	386	9
Windows Phone 8 »	Nokia Lumia 822, HTC 8X and others	320	6
Android 4.0 »	Samsung Galaxy Nexus	297	3
Bada 2.0 »	Samsung Wave and others	283	9
Nokia Belle FP 2 »	560 5.5 Nokia 808 PureView and others	272	9
Android 2.3 »	Google Nexus S and others	200	1

development or beta

		Score	Bonus
Tizen 2 »		492	16
Dolphin Engine Beta »	Android 2.2 or higher	469	3
Opera Mobile 14 »	Android	450	11
Tizen 1 »		426	16
Chrome Beta »	All Android 4 devices	415	11
Nokia Xpress »	Windows Phone	239	2



New HTML5 features on Tizen

HTML5 Feature Set

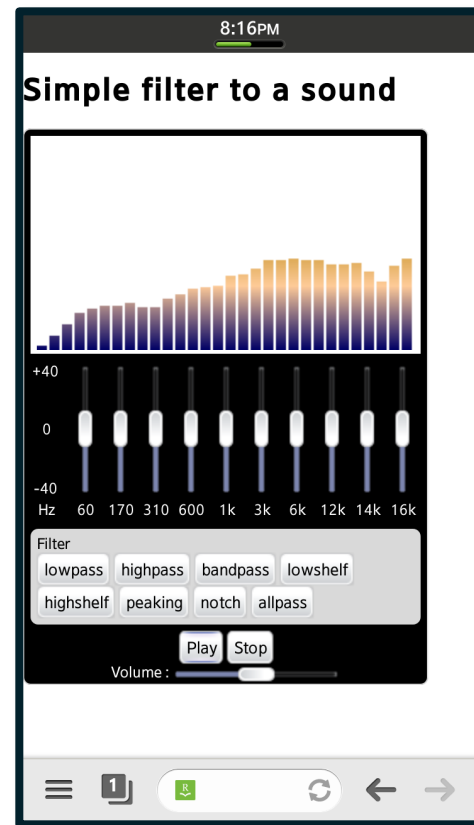
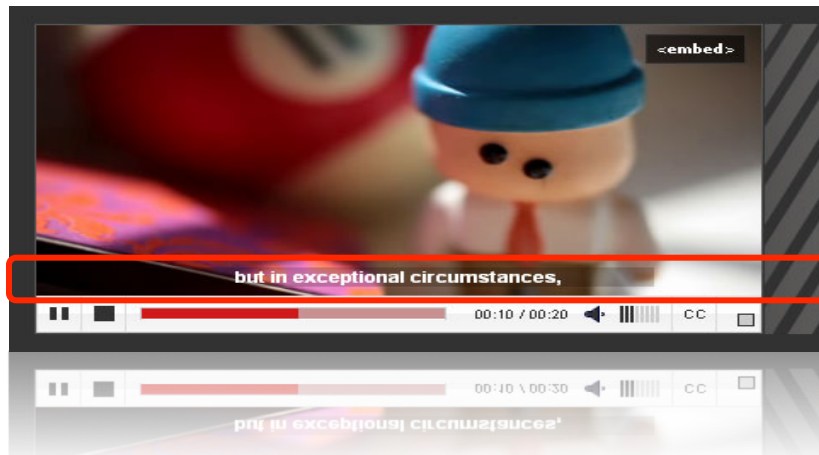
	Specifications♪	Description♪
DOM, Forms and Styles♪	HTML5 Forms , Selectors API, Media Queries, CSS Transforms, CSS Animations, CSS Transitions, CSS Color, CSS Background and Borders, CSS Flexible Box Layout, CSS Text, CSS Basic User Interface, CSS Fonts, WOFF File Format 1.0, DOM	Support fundamental Web document structure and styles
Device♪	Touch Events, Device Orientation, Battery Status API , Vibration API, Browser Online State, Screen Orientation API , Network Information API ♪	Support device capabilities such as sensors, network connectivity information, etc.
Graphics♪	Canvas 2D, SVG	Support Canvas graphic and SVG
Media♪	Video/Audio Element, getUserMedia, Web Audio API , HTML Media Capture ♪	Support Video and Audio capabilities in HTML
Communication♪	Web Socket API, XMLHttpRequest Level2, Session History, Server-Sent Events, Web messaging♪	Support fetching remote resource, communication between client and server, remote event dispatch, etc.
Storage♪	Web Storage, File API, File API: Writer, File API: Directories and System, Application caches, Indexed DB, Web SQL Database♪	Support device storage capabilities

HTML5 Feature Set

	Specifications♪	Description♪
Security♪	Cross-Origin Resource Sharing, iframe sandboxing , Content Security Policy ♪	Support secure mechanism for origin protection and content isolation
UI♪	Clipboard API and Events , Drag and Drop ♪	Support rich user experience
Performance and Optimization♪	Web Workers, Page Visibility, Animation Timing Control , Navigation Timing ♪	Support parallel execution, performance optimization methods
Location♪	Geolocation API♪	Support location watch capabilities

Media

- Web Audio API
- HTML5 Video <track>
- HTML Media Capture



HTML Media Capture (1/2)

- Media Capture refers to **capture attribute** added to <Input> element
- The **capture** attribute enables content authors to **give hints of preferred means to capture local media** such as image, video and sound
- It works by overloading the `<input type="file">` and adding new values for the **accept** attribute

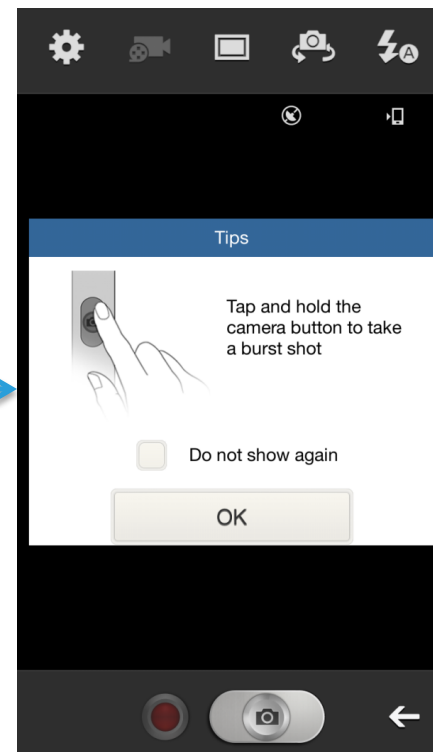
```
<input type="file" accept="image/*" capture="camera">
```


HTML Media Capture (2/2)

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

capture values

camera :	Choose File	No fil...lected
camcorder :	Choose File	No fil...lected
microphone :	Choose File	No fil...lected
filesystem :	Choose File	No fil...lected
null :	Choose File	No fil...lected



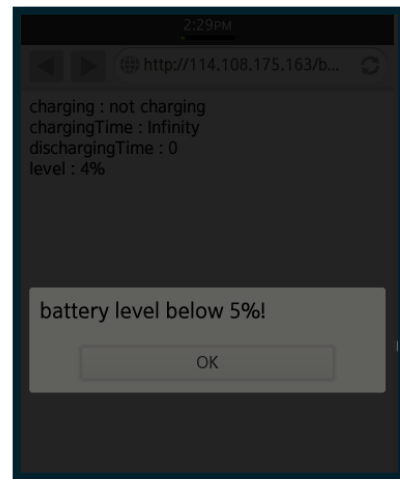
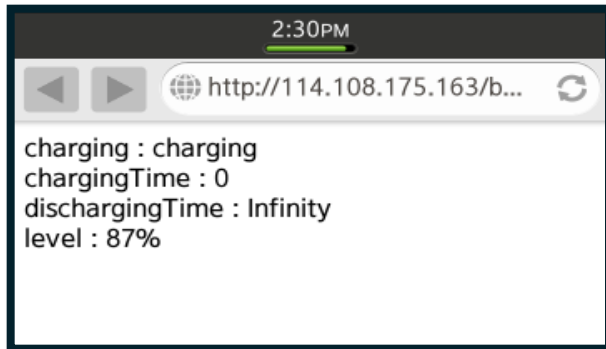
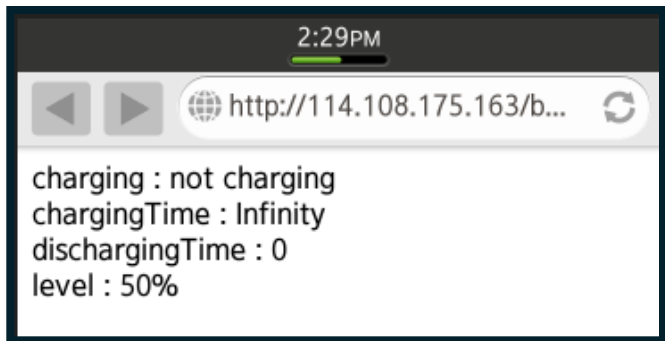
Device

- **Battery Status API**
- **Network Info API**
- **Screen Orientation API**

Battery Status API (1/3)

- The Battery Status API specification defines **a means for web developers to programmatically determine the battery status** of the hosting device
- The Battery Status API can be used to **defer or scale back work when the device is not charging in or is low on battery**
 - An archetype of an advanced web application, **a web-based email client**, may check the server for new email every few seconds if the device is charging, but do so less frequently if the device is not charging or is low on battery
 - Another example is **a web-based word processor** which could monitor the battery level and save changes before the battery runs out to prevent data loss

Battery Status API (2/3)



```
<div>charging : <span id="charging"></span></div>  
<div>chargingTime : <span id="chargingTime"></span></div>  
<div>dischargingTime : <span id="dischargingTime"></span></div>  
<div>level : <span id="level"></span></div>
```

```
<script>  
  // As Tizen is webkit-based, webkit prefix is used.  
  var battery = navigator.battery || navigator.webkitBattery;  
</script>
```

Battery Status API (3/3)

```
<script>
    window.addEventListener('load', function (e) {
        // battery.charging : true or false
        document.querySelector('#charging').textContent =
            battery.charging ? 'charging' : 'not charging';

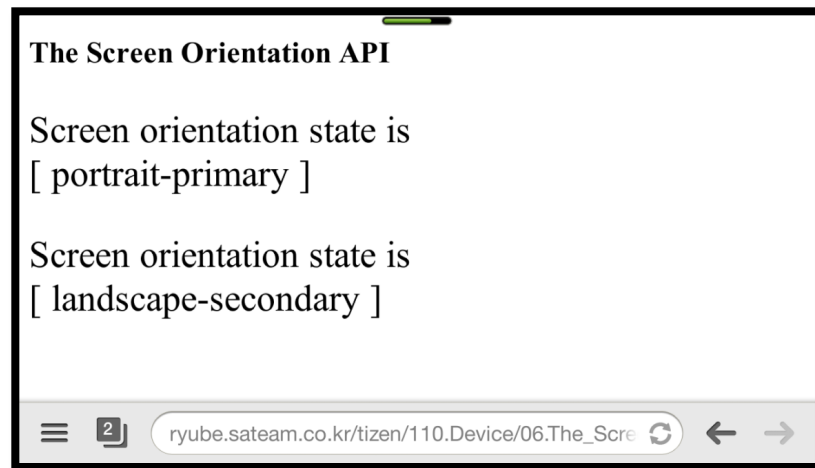
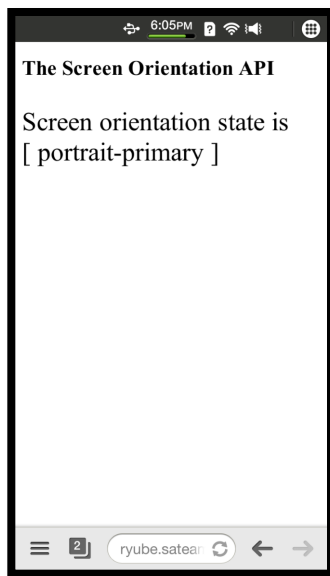
        // battery.chargingTime : second(ex: 3600) or Infinity
        document.querySelector('#chargingTime').textContent = battery.chargingTime / 60;

        // battery.dischargingTime : second(ex: 3600) or Infinity
        document.querySelector('#dischargingTime').textContent = battery.dischargingTime / 60;

        // battery.level : between 0 and 1 (ex: 0.50)
        document.querySelector('#level').textContent = Math.floor(battery.level * 100) + '%';
    }, false);
</script>
```

Screen Orientation API (1/2)

- The Screen Orientation API provides an interface for web applications to **access and lock device screen orientation state**



Screen Orientation API (2/2)

```
<body>
  <h1>The Screen Orientation API</h1>
  <div id="log"></div>
</body>
<script>
  var log = document.getElementById("log");
  function screenState() {
    log.innerHTML += ("<p>Screen orientation state is <br /> [ " + screen.orientation + " ]</p>");
  }
  window.addEventListener("orientationchange", function(e) {
    setTimeout(function() {
      screenState();
    }, 25);
  }, false);
  screenState();
</script>
```

Network Info API (1/2)

- Provides the Network information used in the current device to the Web Application through JavaScript API
- Supports **obtaining the bandwidth of the current network, and checking whether the current network status is paid service**
- **A few hypothetical examples would be:**
 - **Image viewer** showing very low resolution thumbnails
 - **Video game** loading low textures
 - **Email client** downloading only headers or asking the user to download headers
 - Any app trying to aggressively cache any downloaded asset

Network Info API (2/2)

```
<body>
  <h1>The Network Information API</h1>
  <div id="log"></div>
</body>
<script>
  var connection = navigator.connection || navigator.webkitConnection;
  var log = document.getElementById("log");
  function connectionStatus() {
    log.innerHTML = "Connection bandwidth: " + connection.bandwidth + " MB/
s";
    if (connection.metered) {
      log.innerHTML = "The connection is metered!";
    }
  }
  connection.addEventListener("change", connectionStatus);
  connectionStatus();
</script>
```



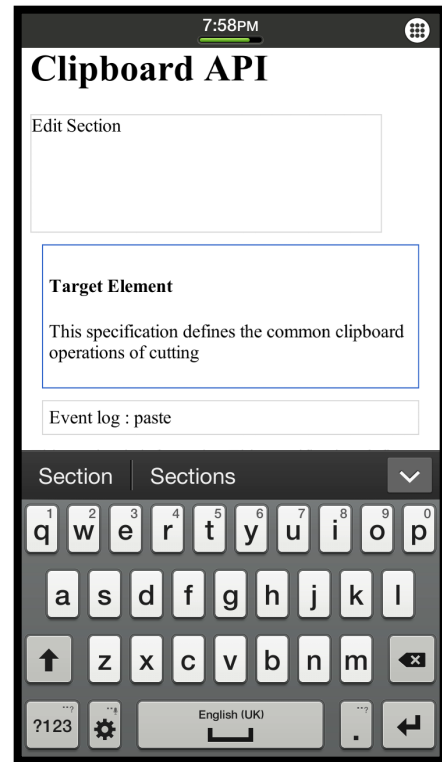
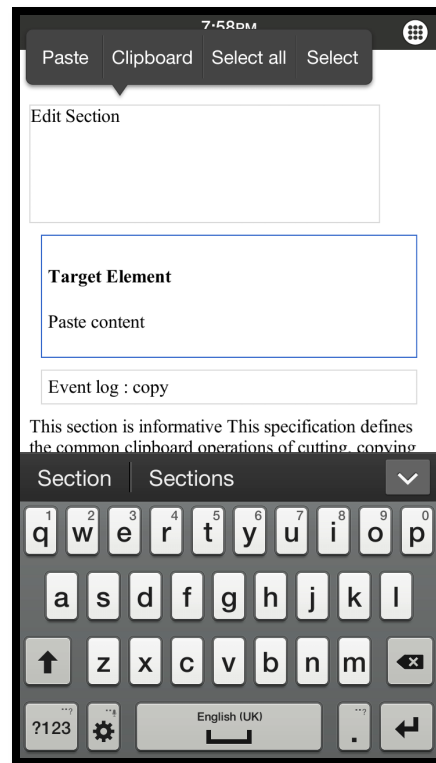
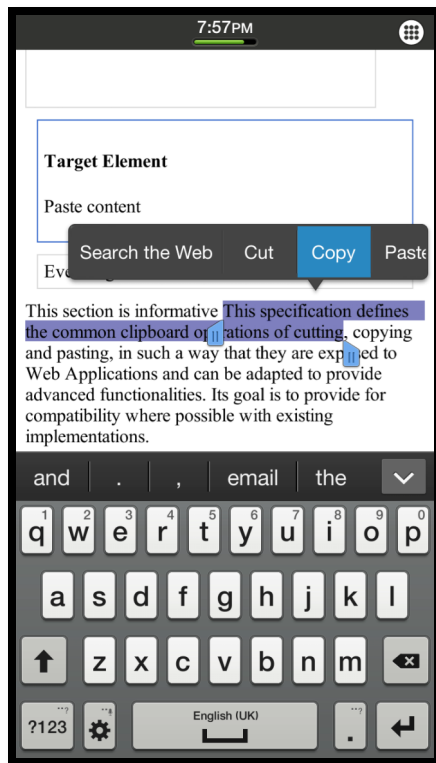
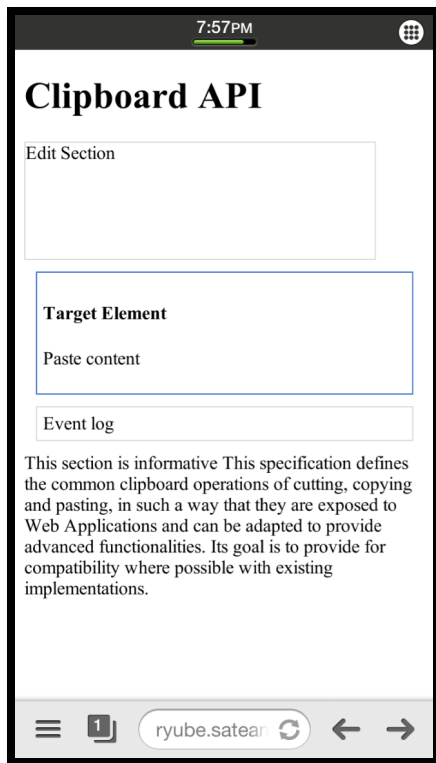
UI

- **Clipboard API**
- **HTML5 Drag and Drop**

Clipboard API (1/5)

- The common **clipboard operations of cutting, copying and pasting**, in such a way that they are exposed to Web Applications and can be adapted to provide advanced functionalities
- Its goal is to provide for compatibility where possible with existing implementations
- **Rich content editing**
 - When copying text which contains hyperlinks or other structure, it is often useful to be able to reformat the content to preserve important information
- **Graphics with built-in semantics**
 - In order to make web applications which allow the manipulation of rich text, or of graphic content such as SVG, it is useful to provide a mechanism that allows for copying more than just the rendered content

Clipboard API (2/5)



Clipboard API (3/5)

```
<style>
  .log {border: 1px solid #d9d9d9; margin: 10px; padding: 5px;}
  .target {border: 1px solid #36c; margin: 10px; padding: 5px;}
</style>
</head>
<body>
  <h1>Clipboard API</h1>
  <div style="width: 300px; height: 100px; border: 1px solid #d9d9d9" contenteditable>Edit Section</div>
  <div class="target">
    <h4>Target Element</h4>
    <p id="target">Paste content</p>
  </div>
  <div id="ev-log" class="log">Event Log</div>
  <div contenteditable>
```

This section is informative

This specification defines the common clipboard operations of cutting, copying and pasting, in such a way that they are exposed to Web Applications and can be adapted to provide advanced functionalities. Its goal is to provide for compatibility where possible with existing implementations.

```
</div>
<body>
```

Clipboard API (4/5)

```
<script>
    var pasteTarget = document.getElementById("target");
    var evLogBox = document.getElementById("ev-log");

    document.addEventListener("cut", function(e) {
        cutHandler(e);
    }, false);

    document.addEventListener("copy", function(e){
        copyHandler(e);
    }, false);

    document.addEventListener("paste", function(e){
        pasteHandler(e);
    }, false);

    function cutHandler(e) {
        evLogBox.innerHTML = "Event log : cut";
    };
</script>
```

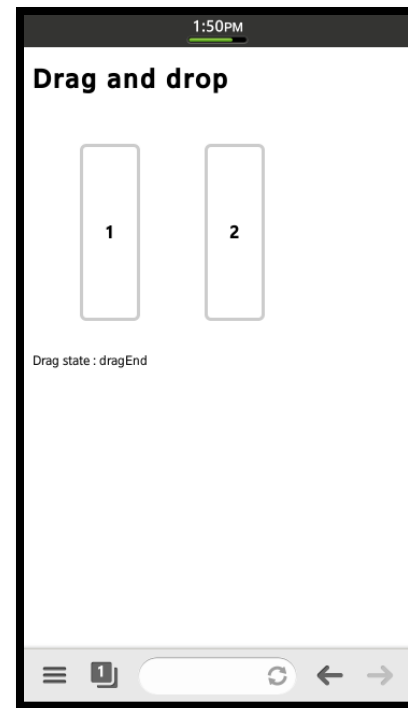
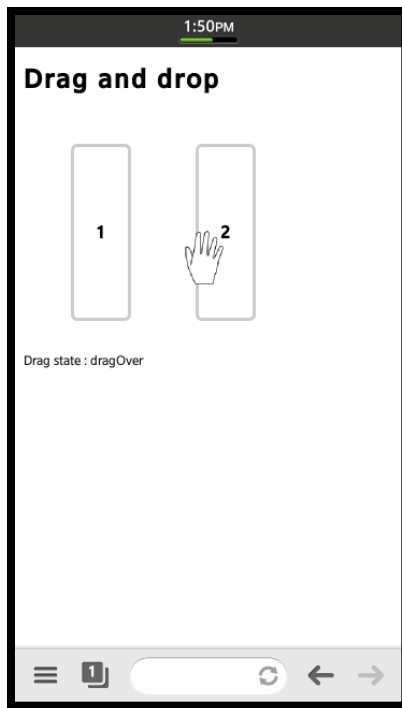
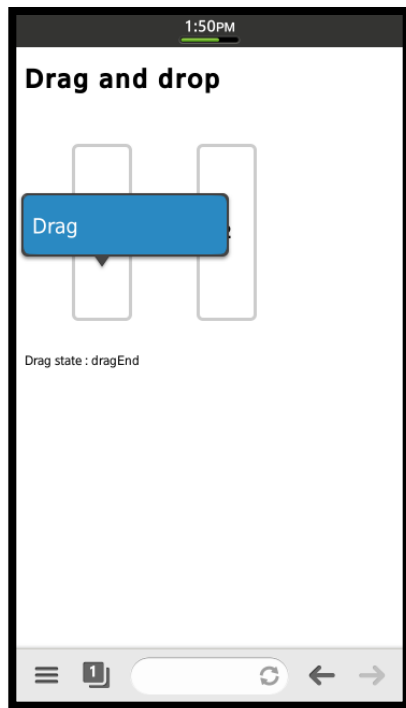
Clipboard API (5/5)

```
function copyHandler(e) {  
    e.preventDefault();  
    // Select area ↵  
    var range = window.getSelection();  
    //store clipboardData↵  
    e.clipboardData.setData("text/plain", range);  
    evLogBox.innerHTML = "Event log : copy";  
};  
function pasteHandler(e) {  
    e.preventDefault();  
    // clipboardData Restore to target  
    pasteTarget.innerHTML = e.clipboardData.getData("text/plain");  
    evLogBox.innerHTML = "Event log : paste";  
};  
</script>
```

HTML5 Drag and Drop (1/5)

- To realize the previous Drag and Drop, JavaScript library was used
- Drag and Drop can be used by **using JavaScript API and the added markup attribute**
- Drag and Drop requires Source, Target, Data and can be used through events
- For events, there are dragstart, drag, dragleave, dragenter, dragover, drop, dragend events, and dragstart and drop event sends data through DataTransfer object

HTML5 Drag and Drop (2/5)



HTML5 Drag and Drop (3/5)

```
<body>
<h1>Drag and drop</h1>
<div class="example_body">
  <div id="drag-list">
    <div class="drag-row" draggable="true">1</div>
    <div class="drag-row">2</div>
  </div>
  <div>Drag state : <span id="log"></span></div>
</div>
</body>
```

HTML5 Drag and Drop (4/5)

```
<script>
    var cols = document.querySelectorAll('#drag-list .drag-row');
    var colsLength = cols.length;
    var log = document.getElementById('log');

    //Set Event Listener in the drag element
    for (var i = 0; i < colsLength; i++ ) {
        cols[i].addEventListener('dragstart', dragStart, false);
        cols[i].addEventListener('drag', dragIng, false);
        cols[i].addEventListener('dragenter', dragEnter, false);
        cols[i].addEventListener('dragover', dragOver, false);
        cols[i].addEventListener('dragleave', dragLeave, false);
        cols[i].addEventListener('drop', dragDrop, false);
        cols[i].addEventListener('dragend', dragEnd, false);
    };
</script>
```

HTML5 Drag and Drop (5/5)

```
<script>
  // Confirm each drag point as log
  function dragStart(e) { log.innerHTML = 'dragStart' };
  function dragIng(e) { log.innerHTML = 'drag' };
  function dragOver(e) {
    e.preventDefault();
    log.innerHTML = 'dragOver'
  };
  function dragEnter(e) { log.innerHTML = 'dragEnter' };
  function dragLeave(e) { log.innerHTML = 'dragLeave' };
  function dragDrop(e) {
    e.stopPropagation();
    log.innerHTML = 'dragDrop'
  }
};
function dragEnd(e) {
  log.innerHTML = 'dragEnd'
};
</script>
```

Performance and Optimization

- `requestAnimationFrame`
- Navigation Timing

requestAnimationFrame (1/3)

- The animation effect of web browser is realized by using CSS3 ([ex. CSS3 animation](#)) JavaScript's `setTimeout`, `setInterval` method
- This kind of animation has the following limitations:
 - Many resources are used to handle complex works such as WebGL, Canvas and so on, so situations occur where the browser stops working
 - As it is impossible to operate accurately to the display frequency of the browser, animation movement sometimes stops midway
 - It is operated in an inactive state of the Web page, wasting resources

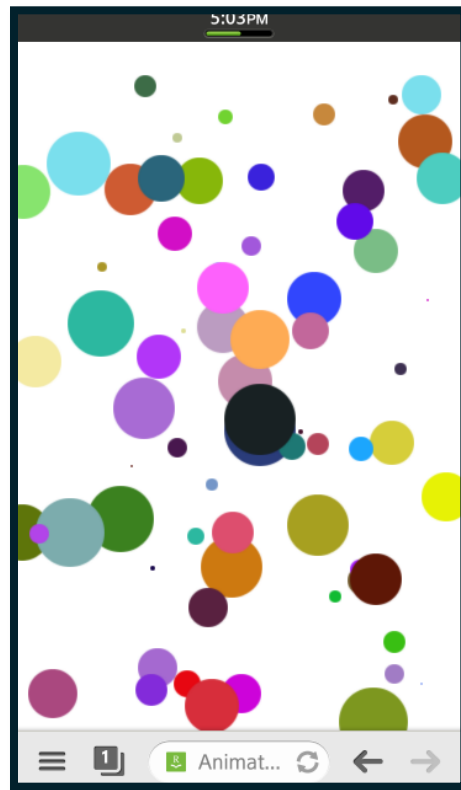
requestAnimationFrame (2/3)

- **requestAnimationFrame** method is a JavaScript API created to resolve such problems
 - `requestAnimationFrame` method is called when the browser updates the page display
 - Animation frame and browser's display frequency matches accurately, so the waste of resource is reduced

requestAnimationFrame (3/3)

- **requestAnimationFrame**
 - Called in case the browser updates the page display
 - Browser display uses 60Hz, which is the same scanning rate of a general monitor

```
function animate() {  
    //Randomly creates coordinate, radius, color  
    var x = Math.floor(Math.random() * canvas.width);  
    var y = Math.floor(Math.random() * canvas.height);  
    var r = Math.floor((Math.random() * 30) + 1);  
    var c = "#" + ((1 << 24) * Math.random() | 0).toString(16);  
    //Creates a circle in random locations  
    context.beginPath();  
    context.arc(x, y, r, 0, Math.PI * 2, true);  
    context.fillStyle = c;  
    context.fill();  
    webkitRequestAnimationFrame(animate);  
}
```





Upcoming cutting-edge HTML5 features

Upcoming HTML5 Features

- Media
 - [Media Source Extensions](#), [Encrypted Media Extensions](#), [WebRTC](#)
- Device APIs
 - [Ambient Light Events](#), [Proximity Events](#)
- Security
 - [Web Cryptography](#)
- Storage
 - [Quota Management API](#)



Thanks for your attention!



TIZEN™

DEVELOPER CONFERENCE

2013

SAN FRANCISCO