Highly portable HTML5 games on TIZEN

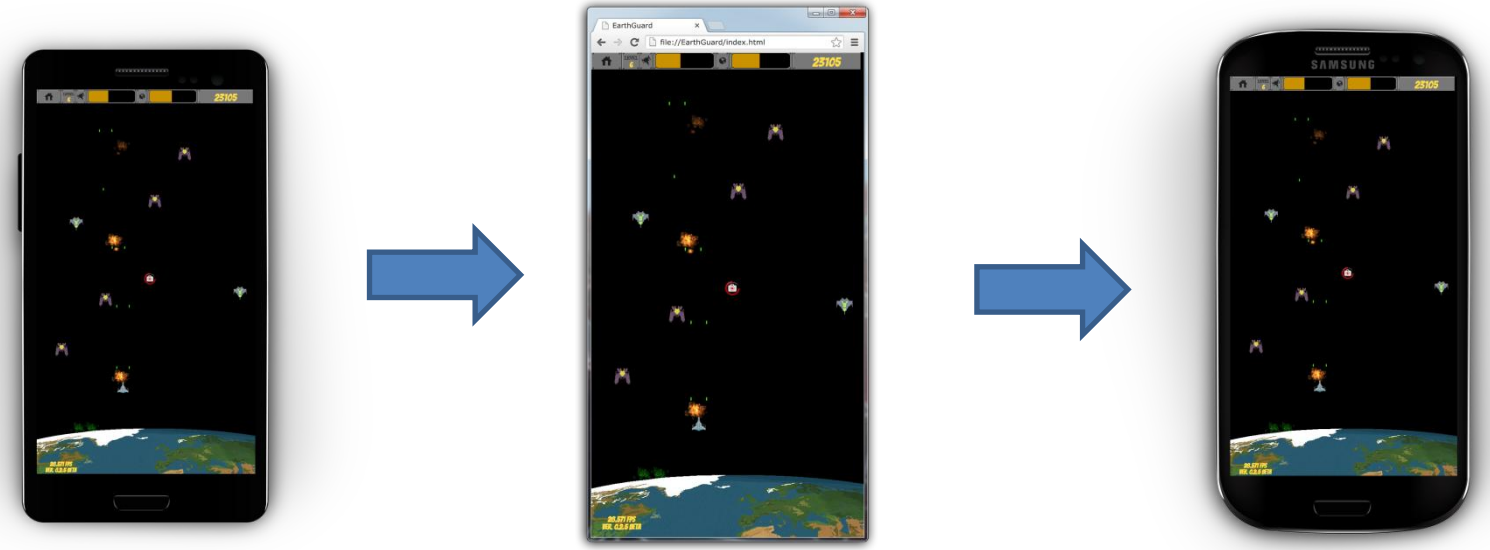
Tizen Developer Conference May 2013
San Francisco USA

Tomasz Ścisło – Software Engineer
Samsung R&D Center Poland

The Earth Guard development story

Code once - for Tizen and deploy your application on HTML5 compatible platforms easily!

You will gain heads up for such a development during this presentation!

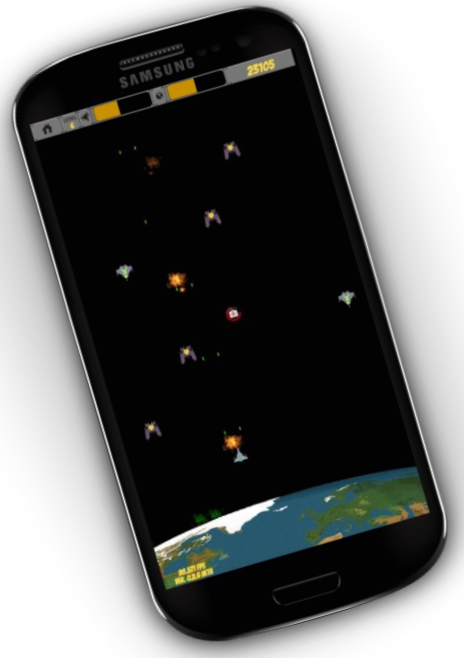


TIZEN

ANDROID

Agenda

- The Earth Guard story
- Initial requirements for highly portable Tizen applications
- Web game concept
- Canvas 2D API introduction
- Responsive design - Adaptive UI
- Handling user input on mobile and on desktop
- Launching on Android platform
- Summary and further reading



Initial requirements for highly portable Tizen applications

Development using open web standards: HTML5, JavaScript, CSS3

Responsive design - adaptive UI

Handling user input on both mobile and desktop

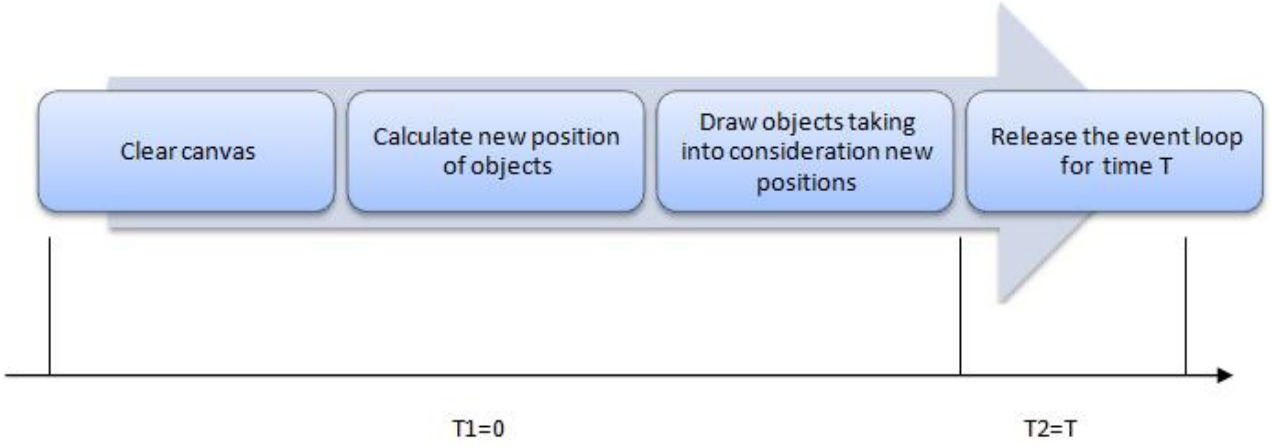
Java wrapper for Android or open source solution: PhoneGap – Cordova usage

Web game concept

- Retained mode vs. Immediate mode
  - Retained mode favors DOM and SVG rendering
  - Immediate mode favors Canvas 2D and WebGL
- The Three Principle Objects:
  - Game
  - Game Board
  - Sprite Sheet
- From game loop to adaptive game loop

Earth Guard:
Immediate mode!
Canvas 2D!

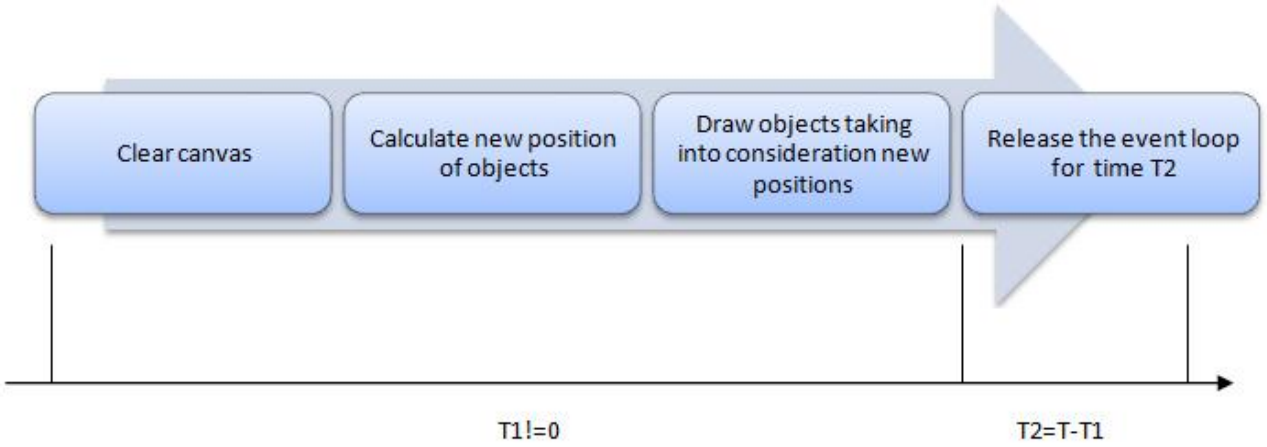
The game loop



FPS = 40
T=1/FPS

T2=1/FPS
T2=25msec

The adaptive game loop



FPS = 40

T2=1/FPS-T1
T2=25msec-T1

Canvas 2D API essentials

Earth Guard uses only 4 Canvas 2D API methods – easy even for beginners!

```
var canvasDOM = document.getElementById('game');
var canvas = canvasDOM.getContext && canvasDOM.getContext('2d');
```

- Handling graphics taken from a sprite sheet

```
canvas.drawImage(image, sx, sy, sWidth, sHeight, dx, dy, dWidth, dHeight);
```

- Filling rectangular box with a solid color

```
canvas.fillStyle = '#000000';  // black
canvas.fillRect(x, y, width, height);
```

- Displaying texts

```
canvas.font = "34px bangers"; // custom font
canvas.fillText('test', x, y, maxWidth);
```

Simple!

Responsive design - adaptive UI

- Canvas scaling – the first step

1. Set maximum resolution for your application – in our case it is HD
2. For any other device's resolution set the canvas height equal to the device height
3. Count scale factor for canvas width
4. Apply this scale for the canvas element using below API call

canvas.scale(x_scale, y_scale)

Responsive design - adaptive UI

- UI scaling – the final step
  - Font scaling

    Reference font = 25px ⇒ body {font-size:25px} ⇒ 100%

    For a device with different resolution than HD we apply the scaling factor for the body element font-size CSS property:

    ```
    var scaledFontSize = Math.round(25 * scale);
    $(document.body).css('fontSize', scaledFontSize + 'px');
    ```

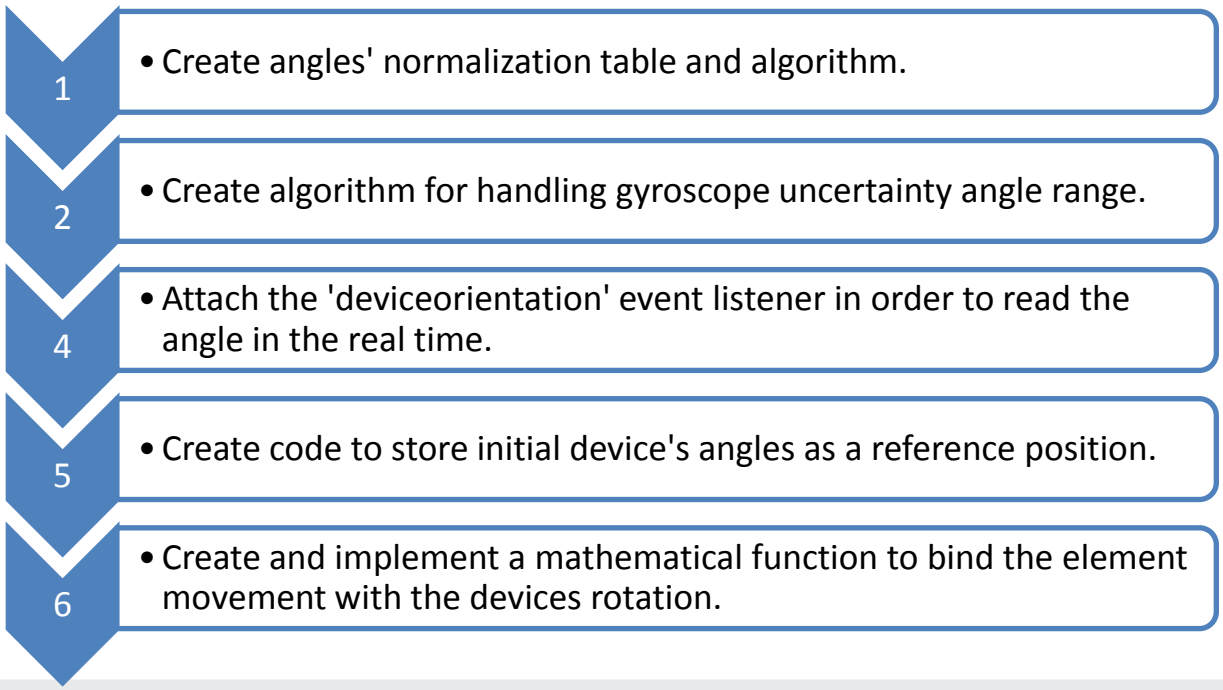
  - UI controls dimension and positions scaling

    All HTML elements dimensions should be defined using em units.

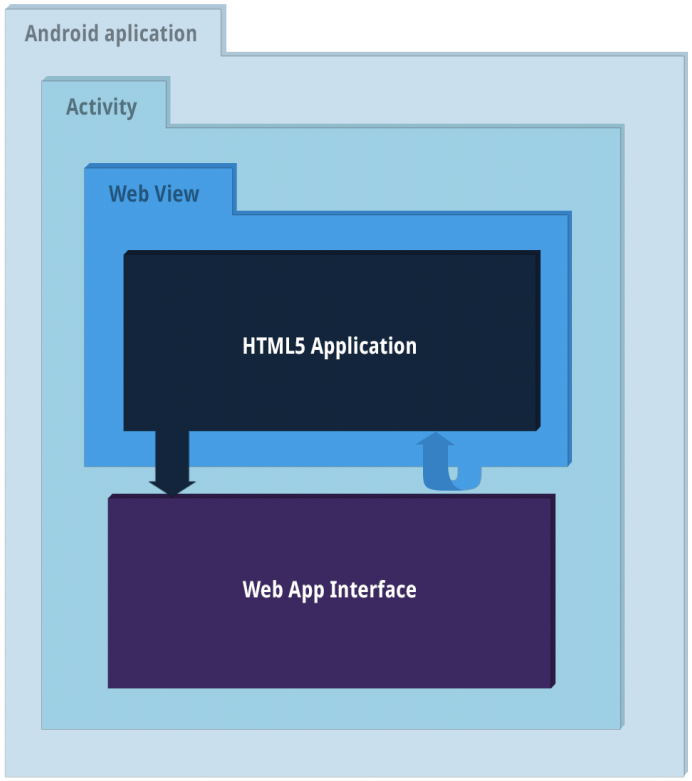
    1em = font size of the element

User input handling on mobile and desktop

- Desktop browser user input types: keyboard, mouse
- Mobile user input types: gyroscope, touch
- For desktop browser bind keydown, keyup, click events
- For mobile bind deviceorientation, touchstart, touchend events



Launching on Android – dedicated Java wrapper



What should you consider when writing your own Java wrapper?

- Providing an interface between the Java wrapper and HTML5 application and vice versa
- Handling console.log messages from the JS application and forwarding them to LogCat
- Handling the back button
- Handling the Android application lifecycle
- Handling Android power management

Interface from Java wrapper to HTML5 application

- Any Java object can be exposed as an namespace in HTML5 application.
- Java class *WebAppInterface*

```
public class WebAppInterface {
…
@JavascriptInterface
public void exitApplication() {
    mActivity.finish();
}
…
}
```

- Interface registration using Activity's build in method – Android namespace registration in HTML5 application:

```
mWebView.addJavascriptInterface(mWebAppInterface, "Android");
```

- API can be used in HTML5 application:

```
Android.exitApplication();
```

Handling console.log

```
mWebView.setWebChromeClient(new WebChromeClient() {
    public boolean onConsoleMessage(ConsoleMessage cm) {
        Log.d(getString(R.string.js_console_tag),
        "[EarthGuard]: " + cm.message() + " -> line(" + cm.lineNumber() + ") of " + cm.sourceId());
        return true;
    }
});
```

Handling Android back button

```
@Override
public boolean onKeyDown(int keyCode, KeyEvent event) {
    if ((keyCode == KeyEvent.KEYCODE_BACK)) {
        mWebView.loadUrl("javascript:menu.toggle()");
        return false;
    }
    return super.onKeyDown(keyCode, event);
}
```

Thanks to SPRC Earth Guard team

- Graphics: Ewa Mazur
- Developers:
  Tadeusz Włodarkiewicz,
  Żaneta Szymańska,
  Łukasz Jagodziński,
  Karolina Królik
- Technical leader: Tomasz Ścisło

Further reading – developer.tizen.org

- **Canvas2D mobile web game development – basics:** https://developer.tizen.org/documentation/articles/canvas2d-mobile-web-game-development-%E2%80%93-basics
- **Canvas2D mobile web game development – implementation:** https://developer.tizen.org/documentation/articles/canvas2d-mobile-web-game-development-%E2%80%93-implementation
- **Keeping high portability of your Tizen web applications:** https://developer.tizen.org/documentation/articles/keeping-high-portability-your-tizen-web-applications-0
- **Launching Tizen applications on Android platform:** https://developer.tizen.org/documentation/articles/launching-tizen-applications-on-android-platform

- **Earth Guard 1.0.3 for Android on Samsung Apps:** http://apps.samsung.com/venus/topApps/topAppsDetail.as?productId=000000557288

Tomasz Ścisło
t.scislo@samsung.com

SAMSUNG

TIZEN