
Introduction to Tizen and Its Architecture

Minsoo Ryu

**Real-Time Computing and Communications Lab.
Hanyang University**

msryu@rtcc.hanyang.ac.kr

Outline

- ❑ Introduction
- ❑ Overview of Tizen Architecture
 - Web Framework
 - Native Framework
- ❑ Conclusion

What is Tizen?

- ❑ Tizen is an open-source operating system based on
 - the Linux kernel and GNU C library
 - HTML5
- ❑ Tizen targets a wide variety of devices
 - Smartphones, tables, IVI (in-vehicle infotainment), smart TVs, wearable devices, home appliances, ...
- ❑ Tizen is a project governed by
 - A TSG (Technical Steering Group) within the Linux Foundation
 - Two major members of the Tizen association are Samsung and Intel



History of Tizen

- ❑ Tizen roots back to
 - the Samsung SLP (Samsung Linux Platform)
 - The LiMo (Linux Mobile) project
- ❑ Samsung's collaboration with the EFL project, and especially Carsten Haitzler, was known as LiMo for years
 - It was renamed Tizen when Intel joined the project in September 2011, after leaving the MeeGo project
 - A common misconception is that Tizen is a continuation of MeeGo
 - In fact, it builds on Samsung Linux Platform (SLP), a reference implementation delivered within LiMo



pda -> pda phone (pda+phone)
phone -> smart phone (phone + pda)
- i-phone
? -
porting
porting optimize 가

History of Tizen

❑ January 2012

- The LiMo Foundation was renamed Tizen Association



❑ In 2013

- Samsung merged Bada into Tizen



❑ October 2013

- Samsung's NX3000M smart camera was the first consumer product based on Tizen

❑ January 2015

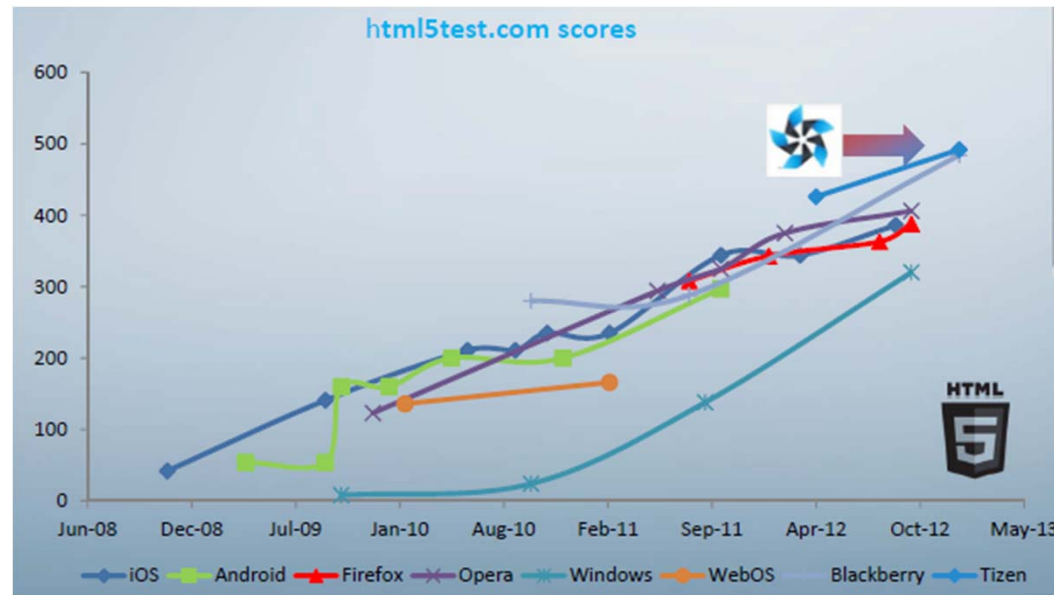
- Samsung released Tizen-based Z1 smartphone to the Indian market



Tizen 2.0

❑ Tizen leads all other mobile platforms in support of HTML5

- Highest on both html5test score and bonus points
 - 492 out of possible 500!
- Receives max bonus points of 16



Tizen Mobile Profile Release History

Tizen 1.0

Apr. 2012

Web-centric platform

- Highest HTML5 coverage
- Tizen Device Web API
- Web UI framework (jQueryMobile based Extension)

Tizen 2.0

Feb. 2013

Web/native dual framework

- Native API
- Unified SDK for Web and native
- Web Runtime based on WebKit2
- Web Audio, HTML Media Capture
- HTML Drag & Drop, Clipboard

Tizen 2.1

May 2013

Hybrid Web/Native, Enhanced Security, and Optimized Perf.

- Hybrid Web and native app support
- Content security policy
- Trusted inter-app sharing
- Account management
- QR code and image recognition
- Systemd replacing init daemon

Tizen 2.2

July 2013

Commercial Ready w/ Enhanced UX

- H/W Menu & Back key
- Better Font Legibility
- H/W LED Notification
- Integration of Apps w/ Contact
- Native API for Secure Element
- UI Customizer
- Live Web App. Editing

Linux kernel 2.6.36

Linux kernel 3.0 (w/ many 3.4 features backported, such as CMA/IOMMU)
Memory optimization for graphics (Framebuffer → DRM/GEM, DMABUF)
eMMC 4.5 support, V4L2 (for codec and camera) support

Tizen 2.x Source Code and SDK Release

- ❑ **Tizen provides application development tools**
 - Based on the JavaScript libraries jQuery and jQuery Mobile

- ❑ **SDK (software development kit) supports HTML5 and related Web technology**
 - oFono is the telephony stack
 - Smack is utilized to sandbox HTML5 web applications
 - **Windowing system**
 - The X Window System with the Enlightenment Foundation Libraries
 - Wayland: Tizen up to 2.x supports Wayland in in-vehicle infotainment (IVI) setups and from 3.0 onward defaults to Wayland
 - **ZYpp was chosen as package management system (PMS)**
 - **ConnMan was chosen over NetworkManager**

Tizen Open Source Information

❑ Visit

- <http://www.tizen.org>
- <http://developer.tizen.org/sdk>
- <http://source.tizen.org/>
- <https://developer.tizen.org/documentation>

❑ Community

- Mailing lists: <http://www.tizen.org/community/mailling-lists>
- IRC Channel: #tizen
- Wiki: <https://www.tizen.org/community/wiki>
- JIRA: <http://bugs.tizen.org>

Source Code Management

❑ Git

- A particularly powerful, flexible, and lowoverhead version control system that makes collaborative development efficient and robust
- <https://review.tizen.org/git/>

❑ Gerrit

- A web-based code review system, facilitating online code reviews for projects using Git version control system
- Gerrit optimizes the code review process, enhancing review quality
- Gerrit simplifies the maintenance of the Gitbased projects, enabling a more centralized use of Git
- <https://review.tizen.org/gerrit>

Tizen OS Bug Tracking

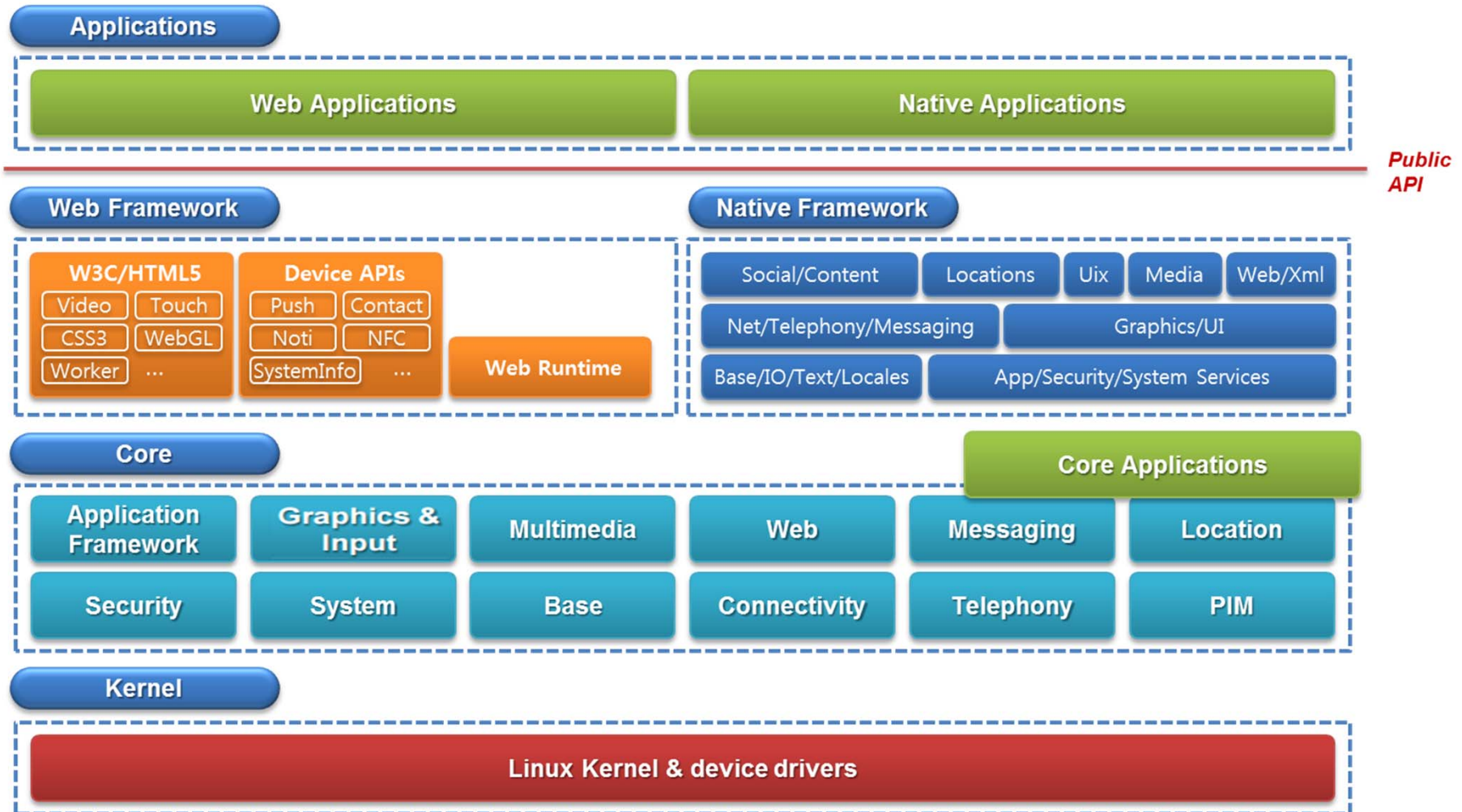
☐ Tizen* uses JIRA to track bugs and to gather feature requests

- <https://bugs.tizen.org/jira/secure/Dashboard.jspa>

☐ Developers need a Tizen account created to

- Add a new bug
- Comment on an existing bug
- Submit a patch to fix bug
- To work on Tizen bug reporting and tracking, a set of guidelines are defined
- <https://www.tizen.org/community/guidelines/bug-guidelines>

The Tizen Architecture (~ v2.2.1)



The Tizen Architecture (~ v2.2.1)

☐ Web framework

- Provides state-of-the-art HTML5/W3C APIs, Web UI framework, supplementary APIs, and additional Tizen device APIs

☐ Native framework

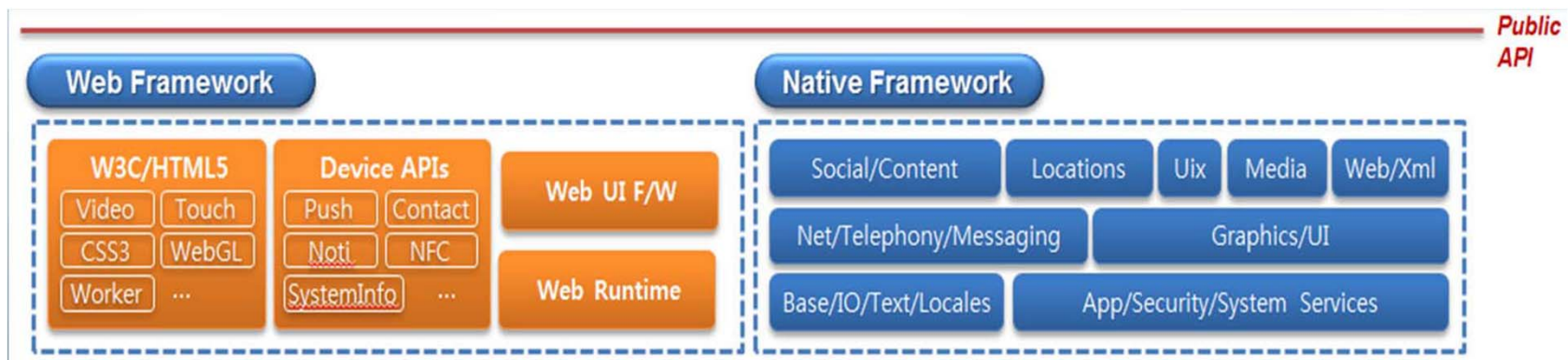
- Supports full-featured native application development and provides a variety of features like background service, image and face recognition, and TTS/STT

☐ Core

- Underlying layer for Web and native providing common functionalities and a security mechanism
- HW adaptation layer with plug-in architecture
- OpenGL® ES/EGL graphics driver

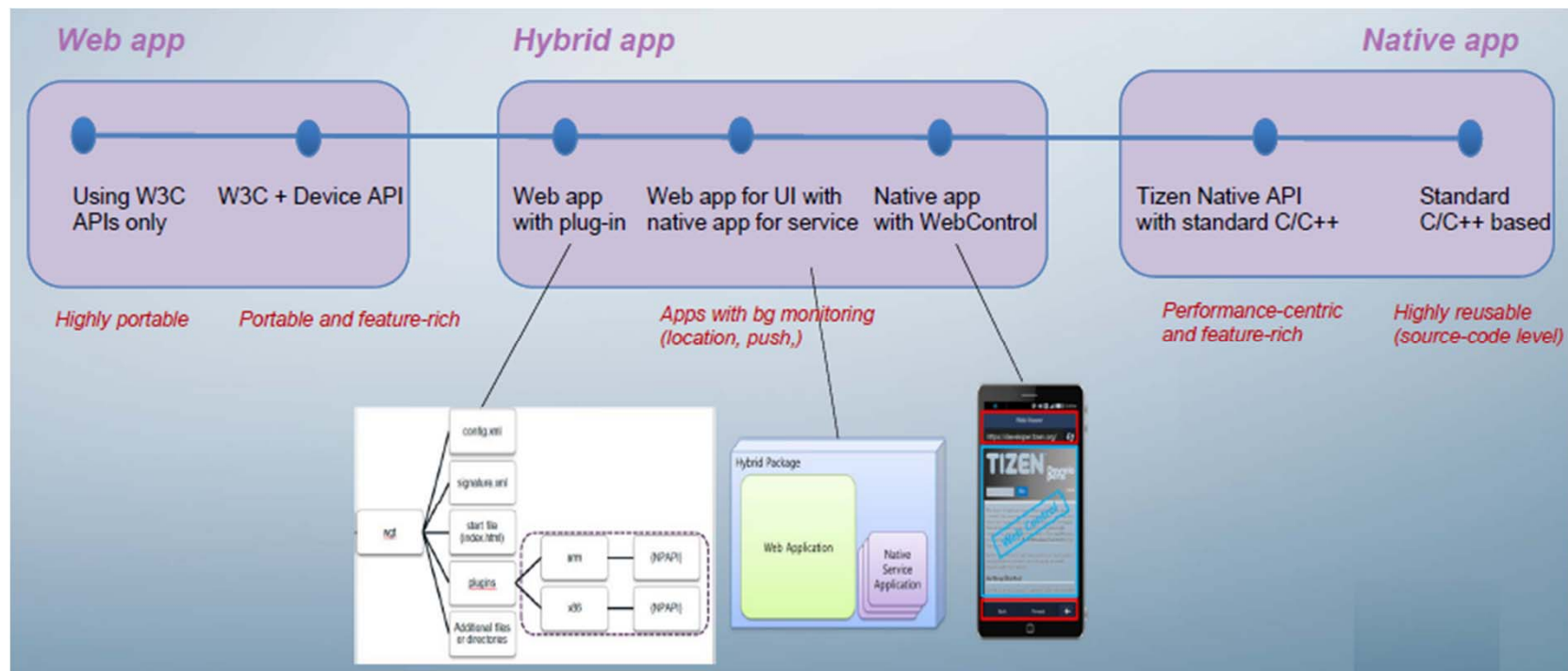
Web vs. Native Framework

- ❑ Native and Web frameworks are complementary to each other
 - Web is strong in portability, ease of app development, and has a minimal learning curve
 - Native is relatively better in terms of performance and memory consumption
 - Native enables reusing the existing engine and libraries written in C & C++ in app development



Web and Native: Mix & Match

- ❑ Different combinations for mixing Web and native, depending on the characteristics or requirements of the app to be developed



Native Framework vs. Core

- ☐ Both are native in nature but focusing on different aspects
- ☐ Core focuses on:
 - Providing common functionalities to Web and native frameworks
 - No need to guarantee app binary compatibility (ABC)
 - Performance and power optimization
- ☐ Native framework focuses on:
 - Application development productivity while guaranteeing ABC
 - Well-documented API references, developer guide, sample codes, and associated tools

Application Types

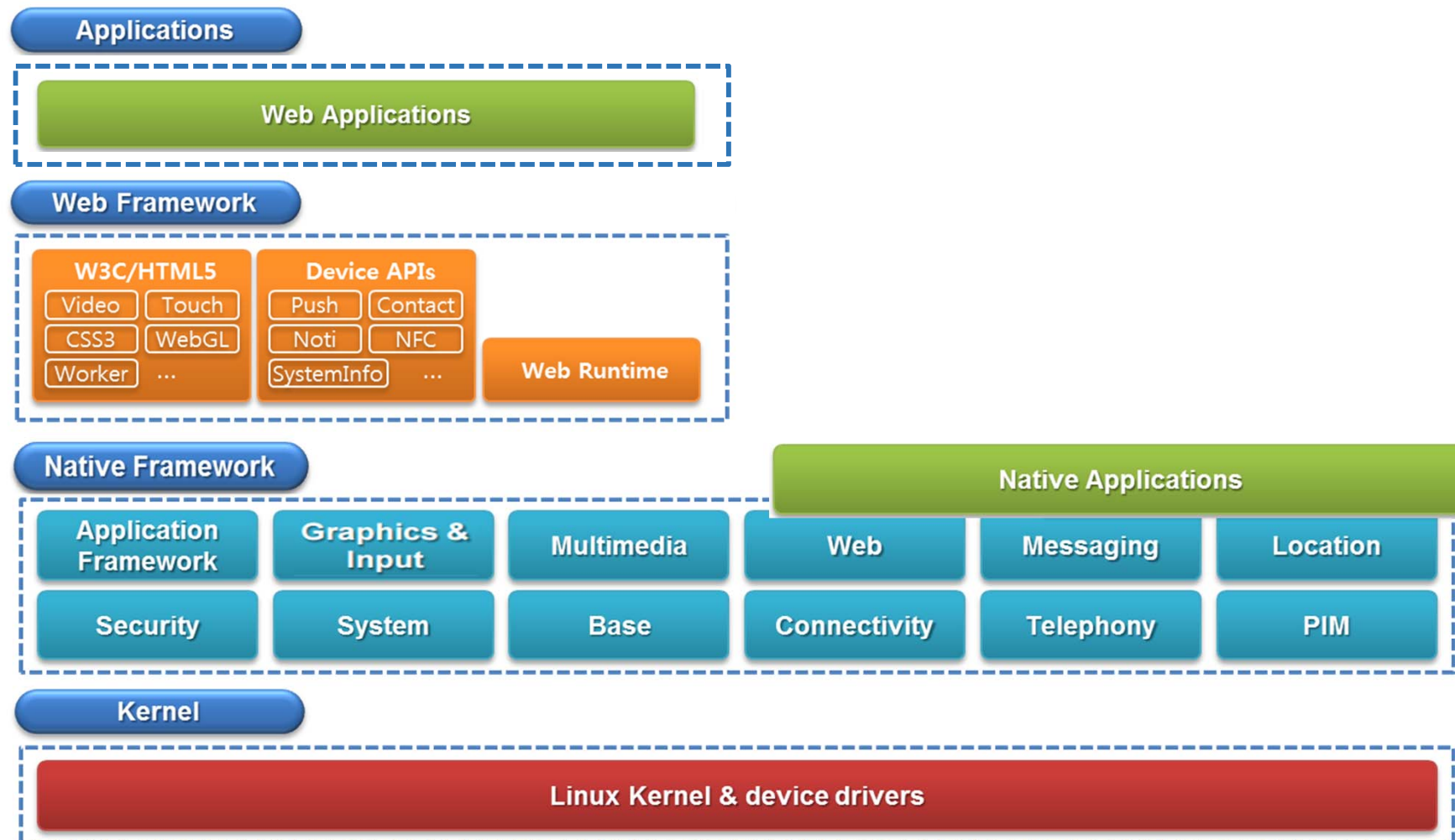
☐ Web and native applications

- Apps using only public APIs to get full support for package installation and upgrade, security, backward compatibility, and so on
- Many sample apps included in the SDK

☐ Core applications

- Apps using Core APIs to fully utilize device capabilities such as telephony
- Usually implemented and preloaded by device implementers
- Backward binary compatibility is not guaranteed

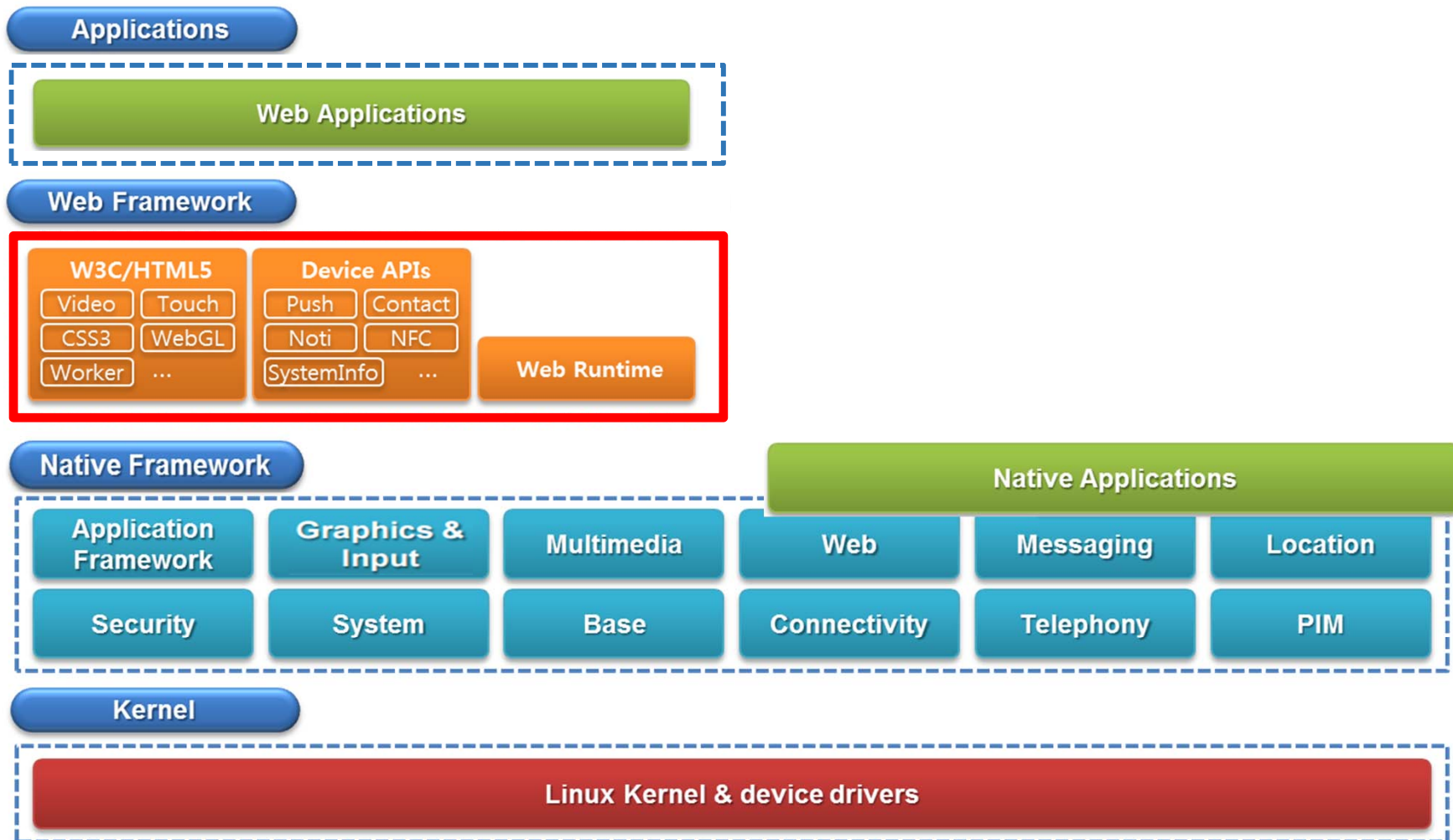
The Tizen Architecture (v2.3 ~)



ARCHITECTURE OVERVIEW

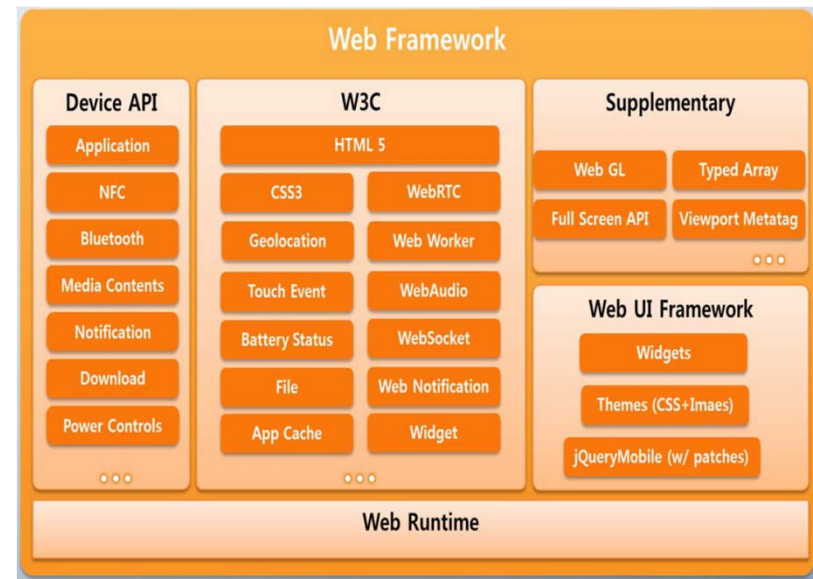
WEB FRAMEWORK

Web Framework



Web Framework

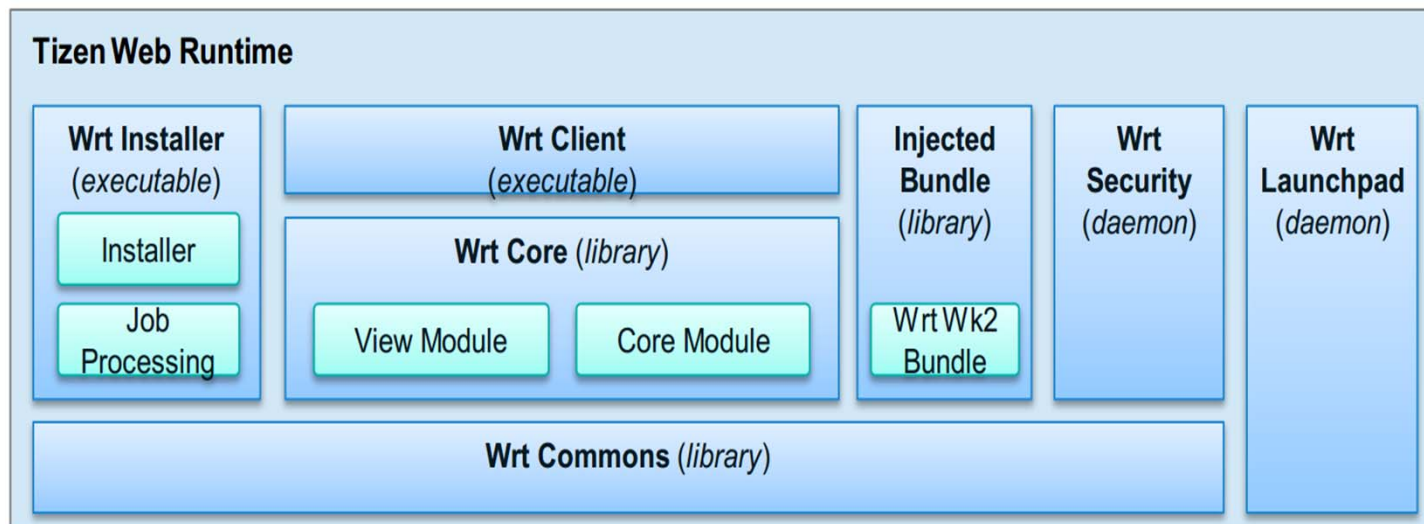
- ❑ **W3C standard Web APIs**
 - W3C/HTML5 markup, CSS, and JavaScript APIs
- ❑ **Supplementary APIs**
 - De-facto APIs (such as Khronos and Mozilla)
- ❑ **Tizen Device APIs**
 - Advanced access to the device's platform capabilities
- ❑ **UI framework**
 - jQueryMobile-based
 - Tools, such as widgets, events, effects, and animations



Web Runtime

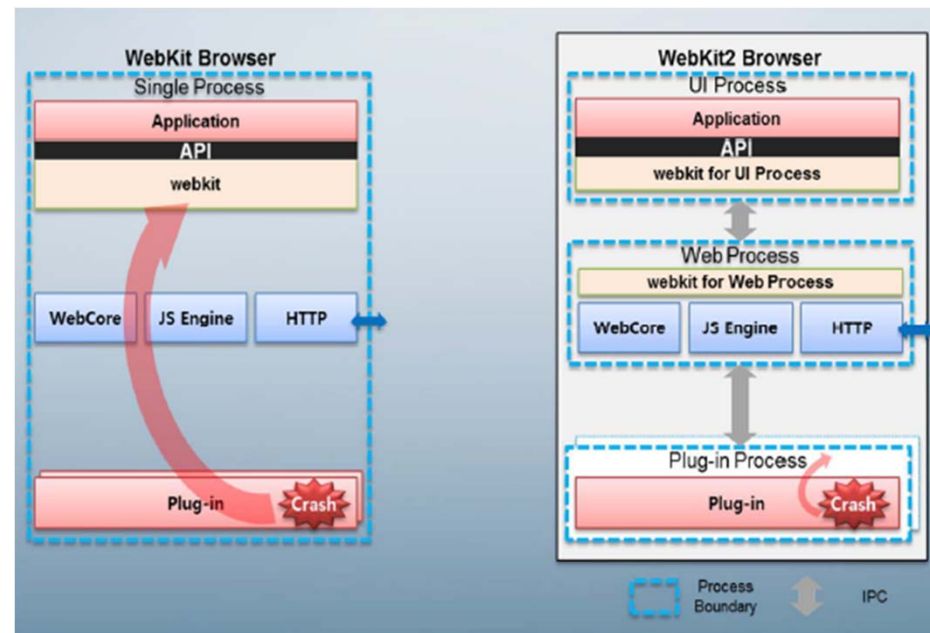
❑ Web Runtime

- Lifecycle Management of web applications
- Execution of web application
- Access to device resources via JS API
- Device and Platform integration
- Access control of web applications



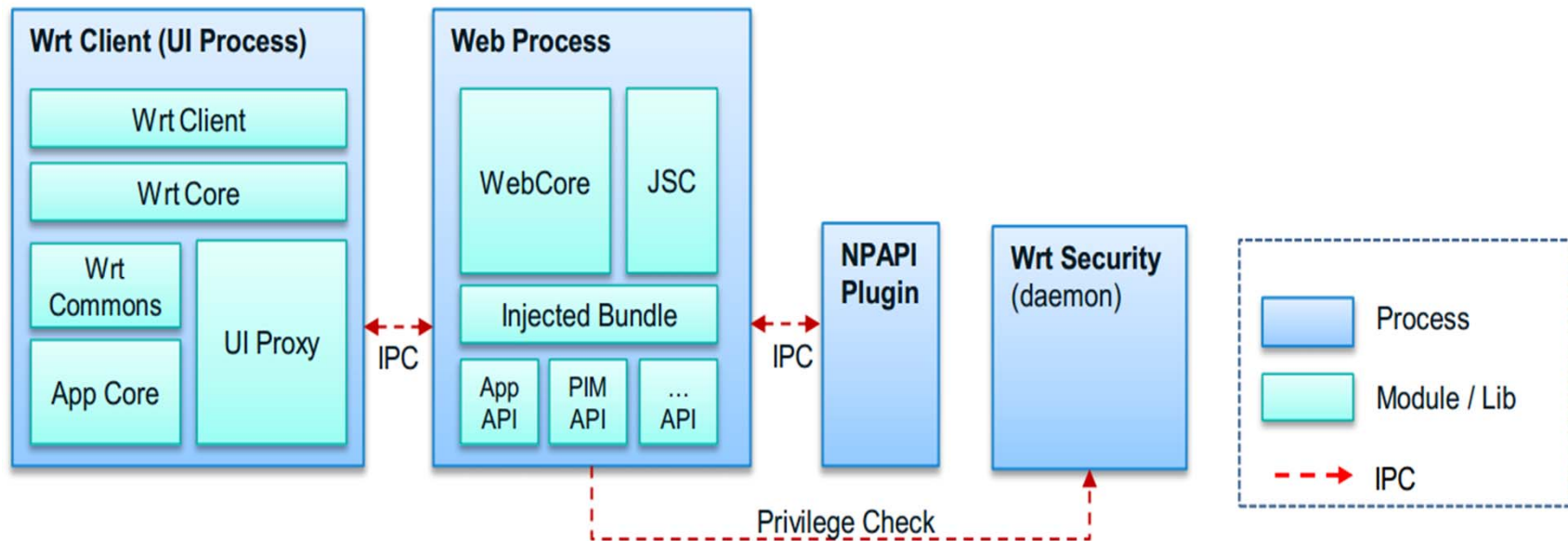
WebKit2 based Browser and Web Runtime

- ❑ Since 2.0, Tizen is using WebKit2 (<http://www.webkit.org>)
 - Split process model for web content and UI with non-blocking APIs
 - UI responsiveness, robustness, security, and better use of multicore CPUs



Web App Process Model

- ❑ Each Web App has 1 UI Process and 1 Web Process
 - UI Process manages lifecycle, and Web Process is responsible for rendering
- ❑ NPAPI (Netscape Plugin Application Programming Interface) plugins will run in separate processes

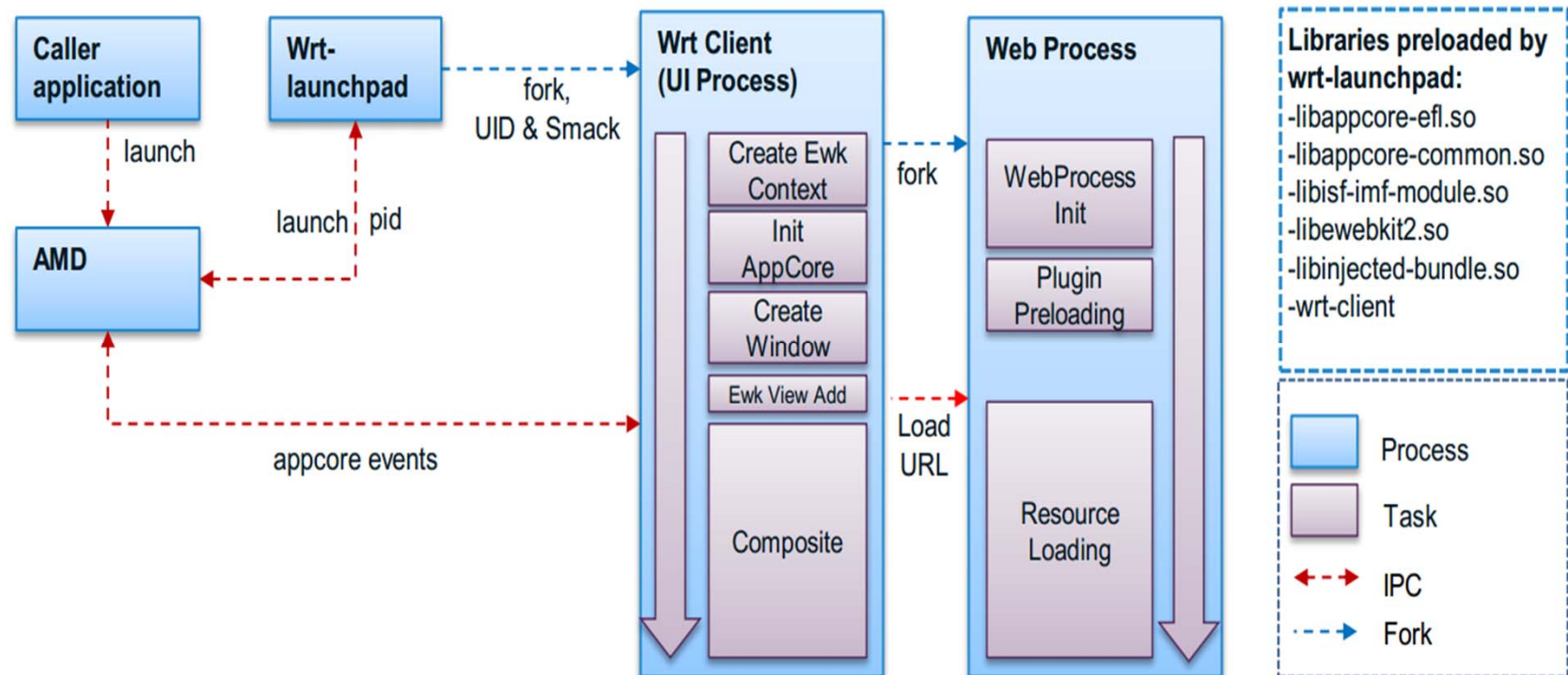


Netscape Plugin Application Programming Interface (NPAPI)

- ❑ A cross-platform plugin architecture used by many web browsers
- ❑ A plugin declares that it handles certain content types (e.g. "audio/mp3")
 - When the browser encounters that content type it loads the associated plugin, sets aside space within the browser context for the plugin to render and then streams data to it
 - The plugin is then responsible for rendering the data
 - The plugin runs in-place within the page, as opposed to older browsers that had to launch an external application to handle unknown content types
- ❑ It was first developed for Netscape browsers, starting in 1995 with Netscape Navigator 2.0,
 - but was subsequently adopted in Internet Explorer 3 in 1996 and implemented by many other browsers, although some browsers later dropped support

Launching Procedure

- ❑ Wrt-launchpad is introduced to preload WebKit and WRT libraries



AMD: Application Manager Daemon

Resource Encryption/Decryption

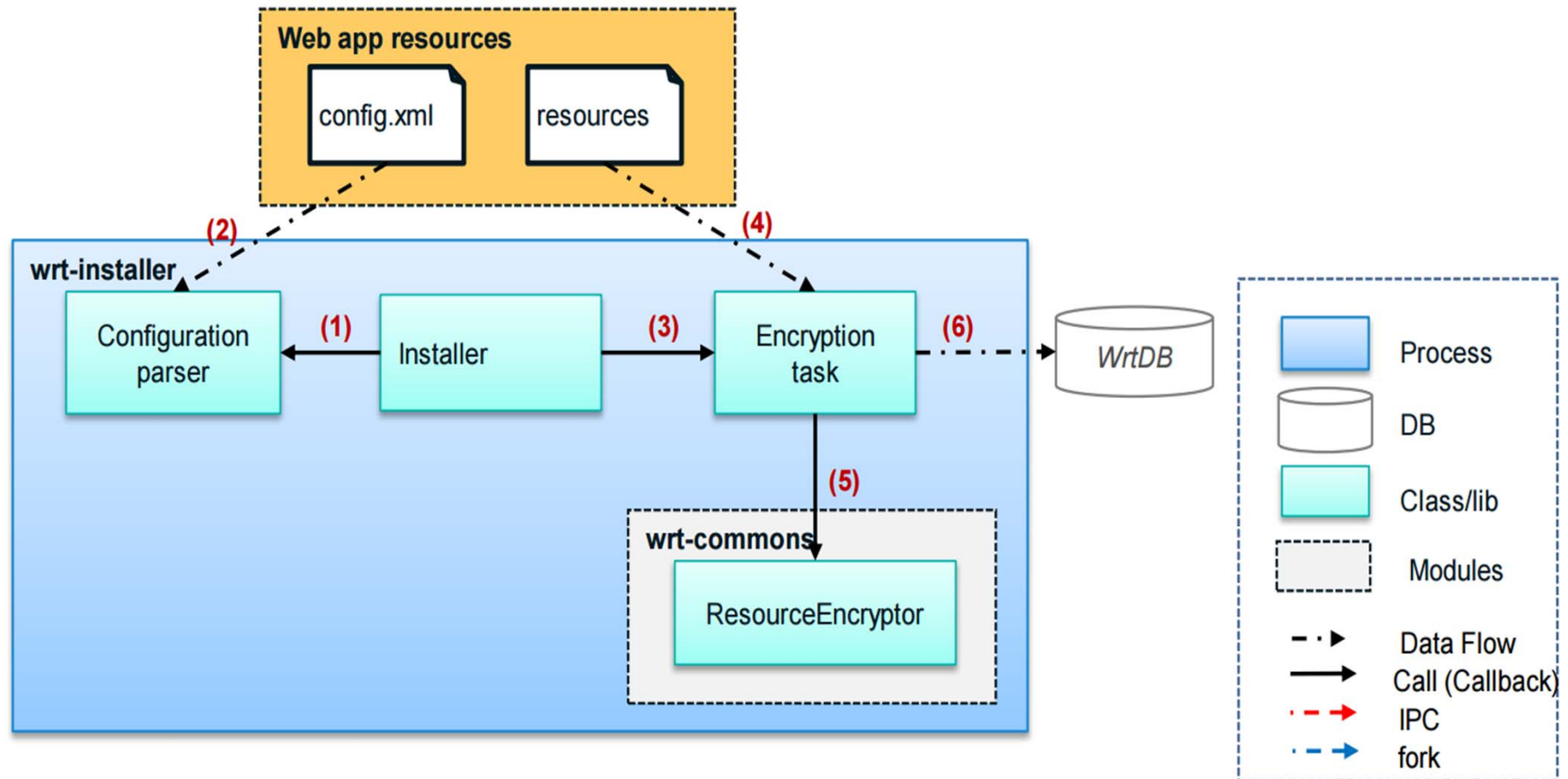
❑ Resources are encrypted during installation / update

- Enabled with `<tizen:setting encryption="enable"/>` in config.xml
- Web App directory is scanned recursively
- Only resources with predefined extensions (html / js / css) are encrypted
- Information about encrypted resources are stored in WRT DB

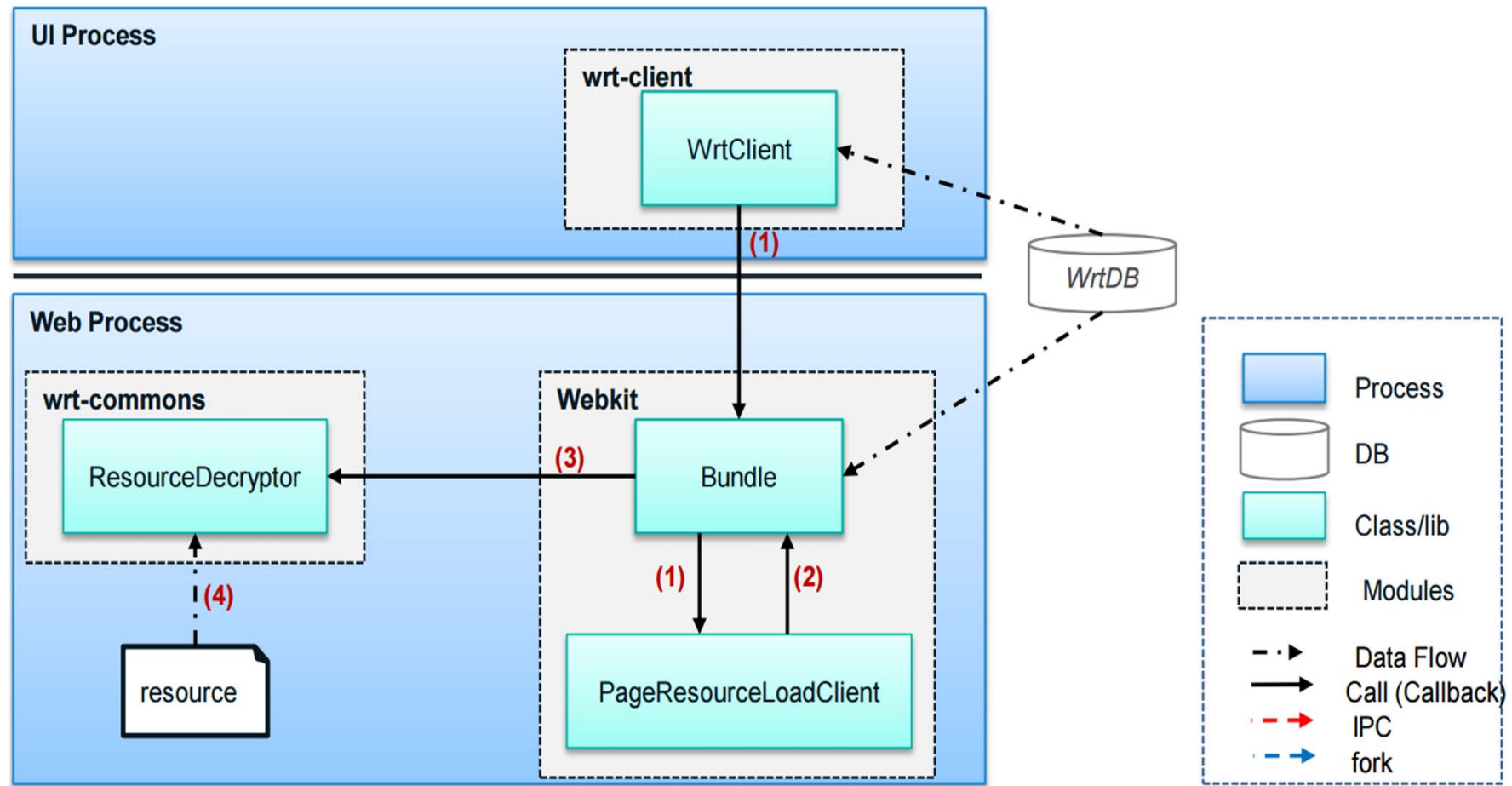
❑ Resources are decrypted at runtime

- UI Process informs Bundle (WebProcess) about the decryption necessity
- Bundle performs resource decryption in *willSendRequestForFrameCallback*
- Resources are decrypted to base64 string and read by WebKit

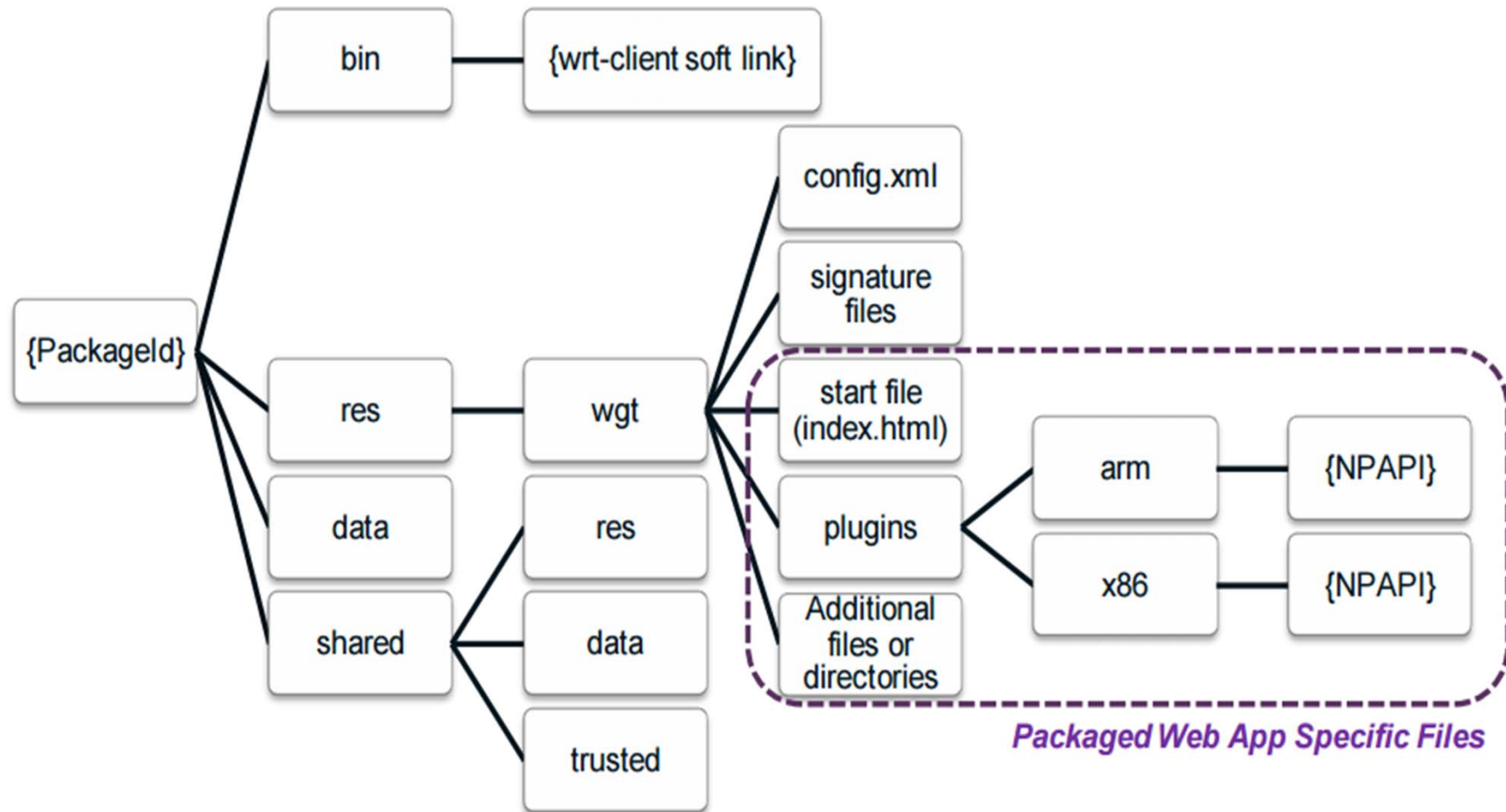
Resource Encryption at Install/Update Time



Resource Decryption at Loading Time



Typical Web App Directory Structure



Web Application Example

□ <head>

- Load resources (css, js etc)

```
<meta name="viewport" content="width=device-width,user-scalable=no"/>
<script src="./lib/jquery.js"></script>
<script type="text/javascript" src="./lib/tau/mobile/js/tau.js" data-build-remove="false"></script>
<link rel="stylesheet" href="./lib/tau/mobile/theme/default/tau.css">

<script src="./js/main.js"></script>

<title>Tizen Web IDE - Template - Tizen - Tizen Web UI Framework - Single-Page</title>
<link rel="stylesheet" type="text/css" href="./css/style.css"/>
```


□ <body>

- Screen layout
- Dispose widgets

```
<div data-role="header" data-position="fixed">
  <h1>System Info </h1>
  <input type="button" value="Check" onClick="buttonCheck()">
</div><!-- /header -->

<div data-role="content">
  <h4>CPU</h4>
  <span id="cpu" style="color: #555;">.</span>
  <h4>Battery</h4>
  <span id="battery" style="color: #555;">.</span>
  <h4>Storage</h4>
  <span id="storage" style="color: #555;">.</span>
</div><!-- /content -->
```

Implemented
in JavaScript



Web Application Example

□ main.js

- Program functions
- Event handler
- Access device info through web device API
- Example: back button event

```
var backEvent = function(e) {  
    if ( e.keyName == "back" ) {  
        try {  
            if ( $.mobile.urlHistory.activeIndex <= 0 ) {  
                // if first page, terminate app  
                unregister();  
            } else {  
                // move previous page  
                $.mobile.urlHistory.activeIndex -= 1;  
                $.mobile.urlHistory.clearForward();  
                window.history.back();  
            }  
        } catch( ex ) {  
            unregister();  
        }  
    }  
}
```


Web Application Example

❑ Retrieve System info

- TAU (Tizen Advanced UI library) of web UI framework
- ./framework/web/web-ui-fw/tau/src/js/core/support/tizen.js
- `getSystemProperty()`

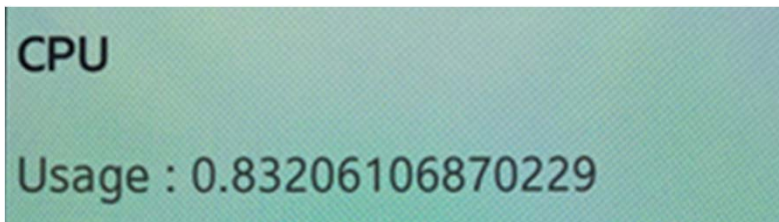
```
function getSystemProperty(property, onSuccess) {  
    try {  
        tizen.systeminfo.getPropertyValue(property, onSuccess, onError);  
    } catch (e) {  
        alert("Exception: " + e.message);  
    }  
}
```

Web Application Example

❑ Display System info

- JavaScript functions
- onSuccess()

```
function onCpuInfoSuccess(cpu) {  
    var message = "Usage : " + cpu.load;  
    document.getElementById("cpu").innerHTML = message;  
}
```



Web Device APIs Categories

Application	
Alarm	Schedules an application to be launched at a specific time
Application	Provides information about applications and controls applications
Package	Provides information about packages and install/uninstall packages
Communication	
Bluetooth	Manages Bluetooth device and supports RFCOMM and HDP
Messaging	Sends and receives SMS, MMS and Email message
NFC	Manages NFC device and detects NFC tag and peer
Push	Receives push notifications from push server
Content	
Content	Discovers multimedia content (such as images, videos or music)
Download	Downloads remote objects by HTTP request
Input/Output	
Filesystem	Accesses the file system and manages file storage
Message Port	Communicates with other applications
Social	
Calendar	Manages events and tasks
Call History	Manages call history for cellular and VoIP calls.
Contact	Manages contacts in device address books
System	
Power	Controls power resources
System Information	Provides information about display, network, storage and other capabilities
System Setting	Sets or gets the system setting values
Time	Provides information about date, time and time zones and manages system time
User Interface	
Notification	Provides a way to notify the user of events that happen in the application

Web Device API's Privileges

❑ API Privilege for higher security

- Privilege level: apps need to have a same or higher privilege level than the API required (Public < Partner < Platform)
- API Privileges need to be described in config.xml

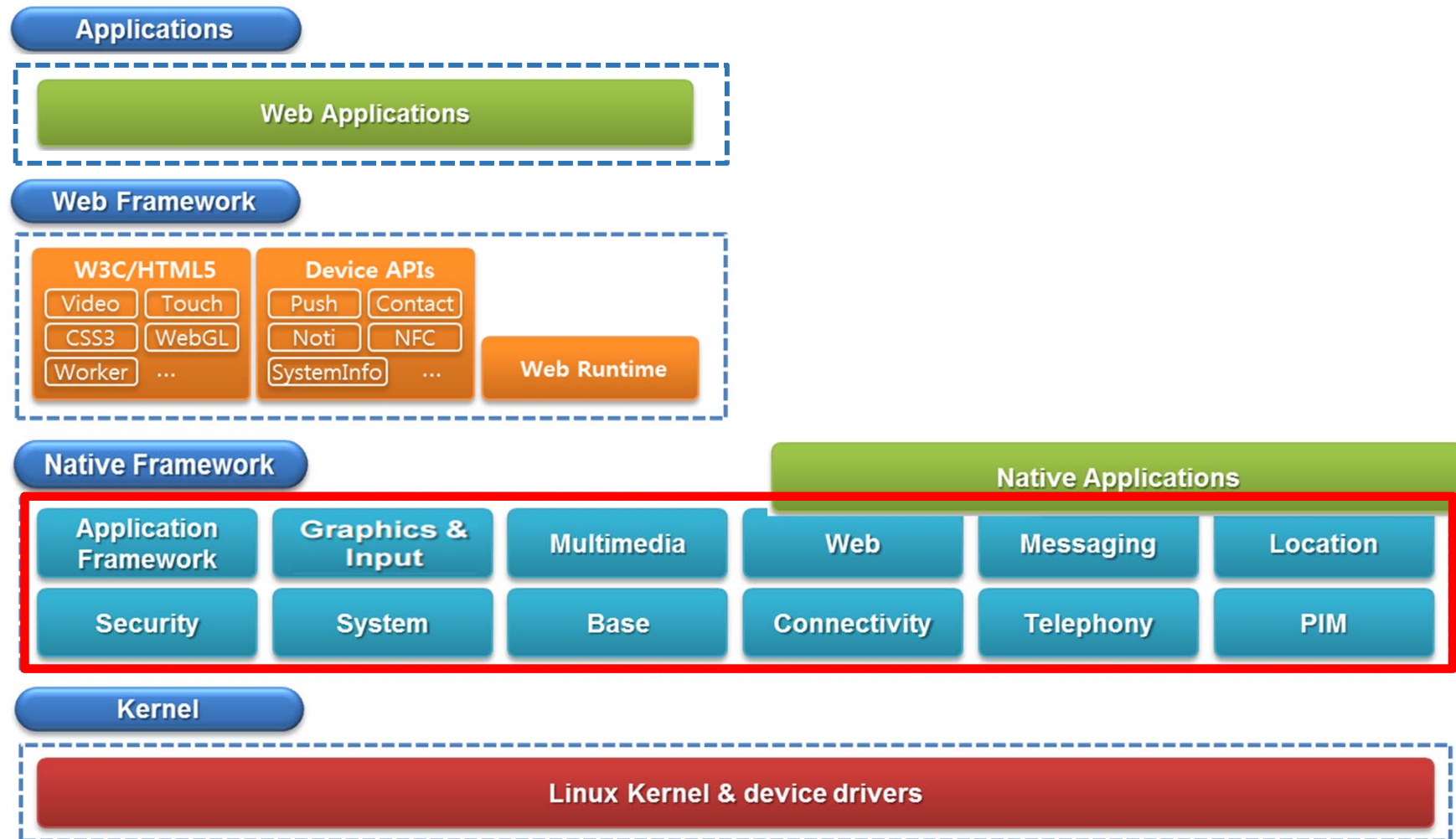
The screenshot displays the Tizen SDK interface. On the left, the 'Tizen Device API Reference' window shows the 'launch' API. It includes a description: 'Launches an application with the given application ID.' and a code snippet: `void launch(ApplicationId id, optional SuccessCallback? success, optional errorCallback? errorCallback);`. Below the code, it states 'Since: 2.0' and lists error types for `ErrorCallback()`: `NotFoundError`, `InvalidValuesError`, and `UnknownError`. At the bottom, a red box highlights the 'Privilege level: public' and 'Privilege: http://tizen.org/privilege/application.launch'.

On the right, the 'Privileges' window is open, showing a list of privileges. The 'Add privilege' dialog is also open, showing a list of internal privileges. The 'http://tizen.org/privilege/application.launch' privilege is selected in the list. The dialog has fields for 'Internal:', 'Privilege name:', and 'File:', with 'Cancel' and 'Finish' buttons at the bottom.

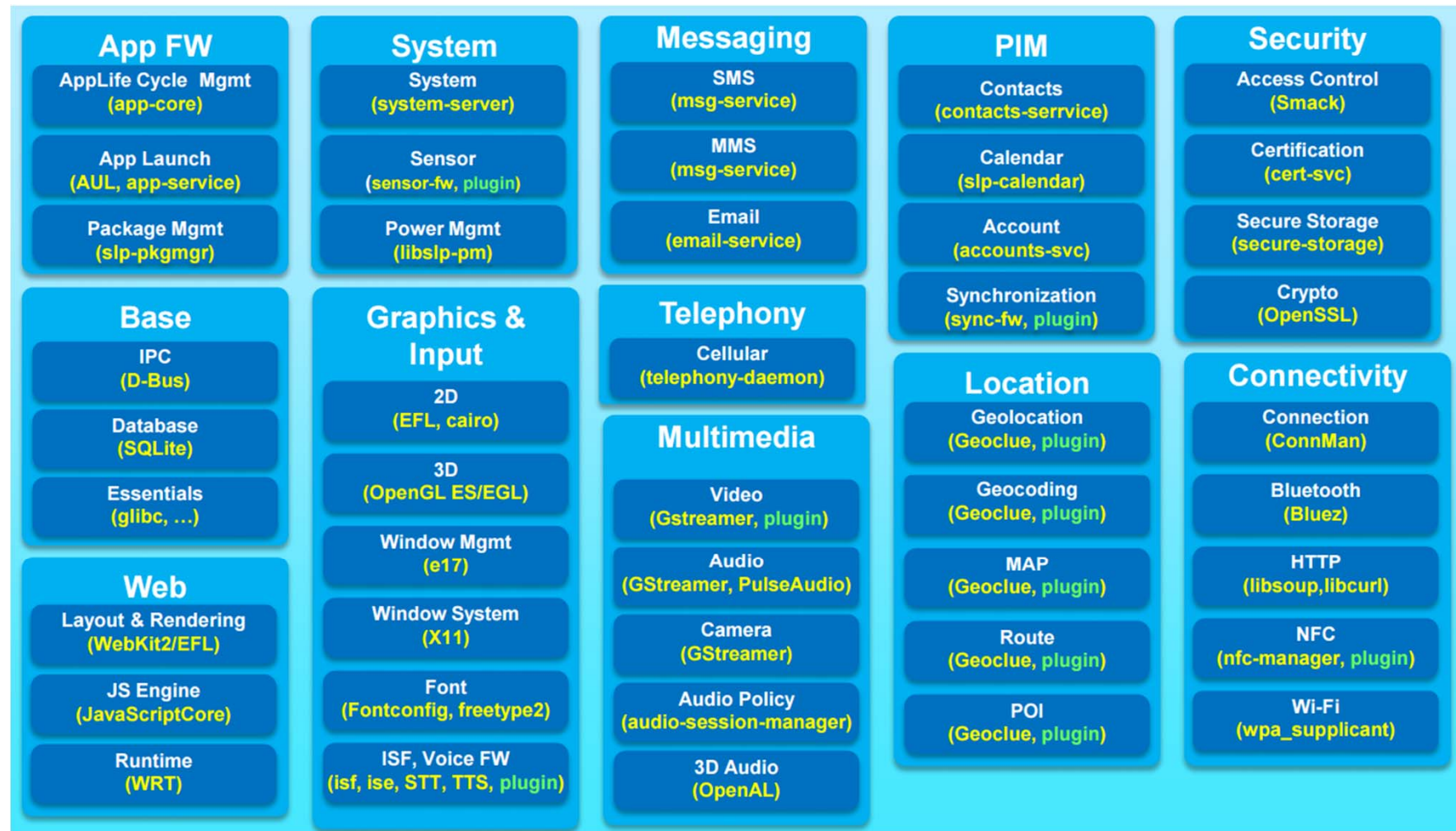
ARCHITECTURE OVERVIEW

NATIVE FRAMEWORK

Architecture Overview



Architecture Overview



Application Framework



□ Provides

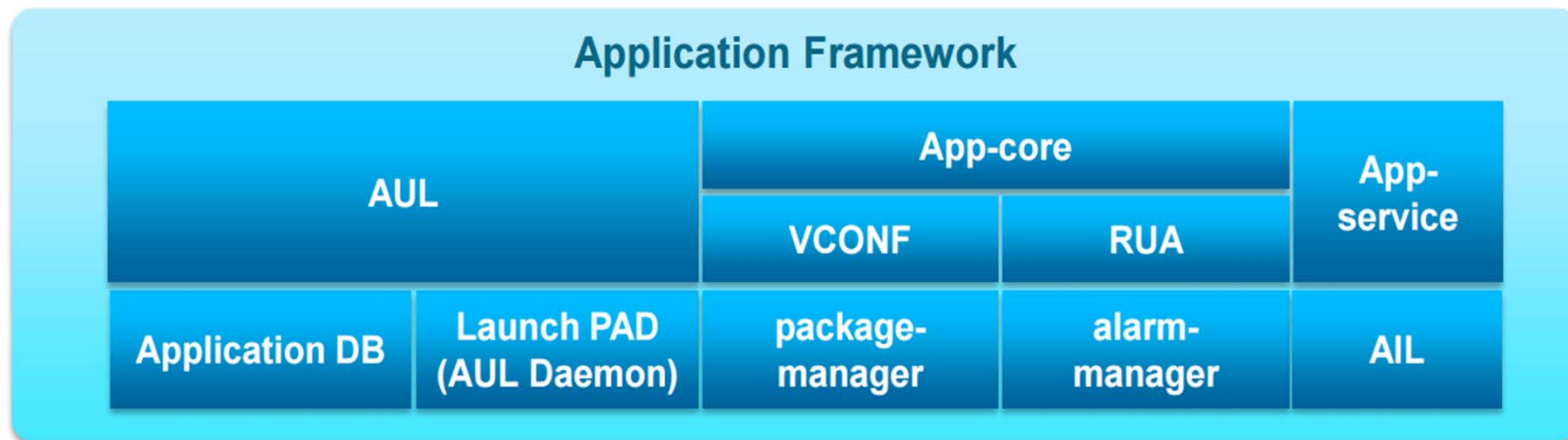
- Launching Application (aul, app-svc)
 - Explicit or implicit information (Combination of Action, URI, and MIME) can be used to determine an app to launch
 - Allowed to launch different type of app (i.e. Web to Native and Native to Web)
- Application life cycle management and handling system events (app-core)
 - Getting app state change notification or system events through main loop
 - Then, calling registered callbacks for the events

Application Framework



□ Provides

- Installing/Uninstalling application (package manager)
- Managing application launched history (librua)
- Setting an alarm to launch at specific time (alarm-manager)



Application Life Cycle

Application Framework	Graphics & Input	Multimedia	Web	Messaging	Location
Security	System	Base	Connectivity	Telephony	PIM

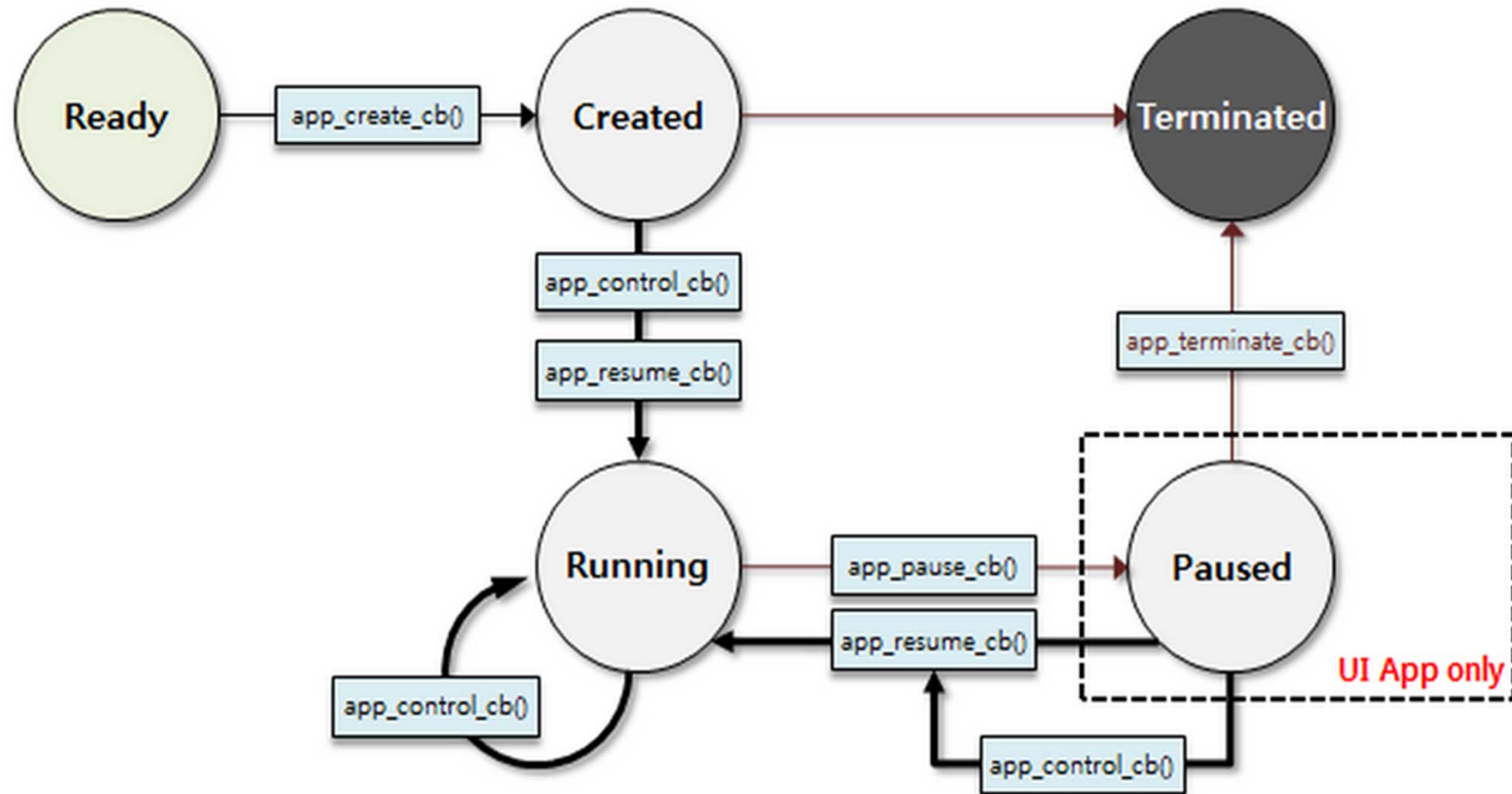
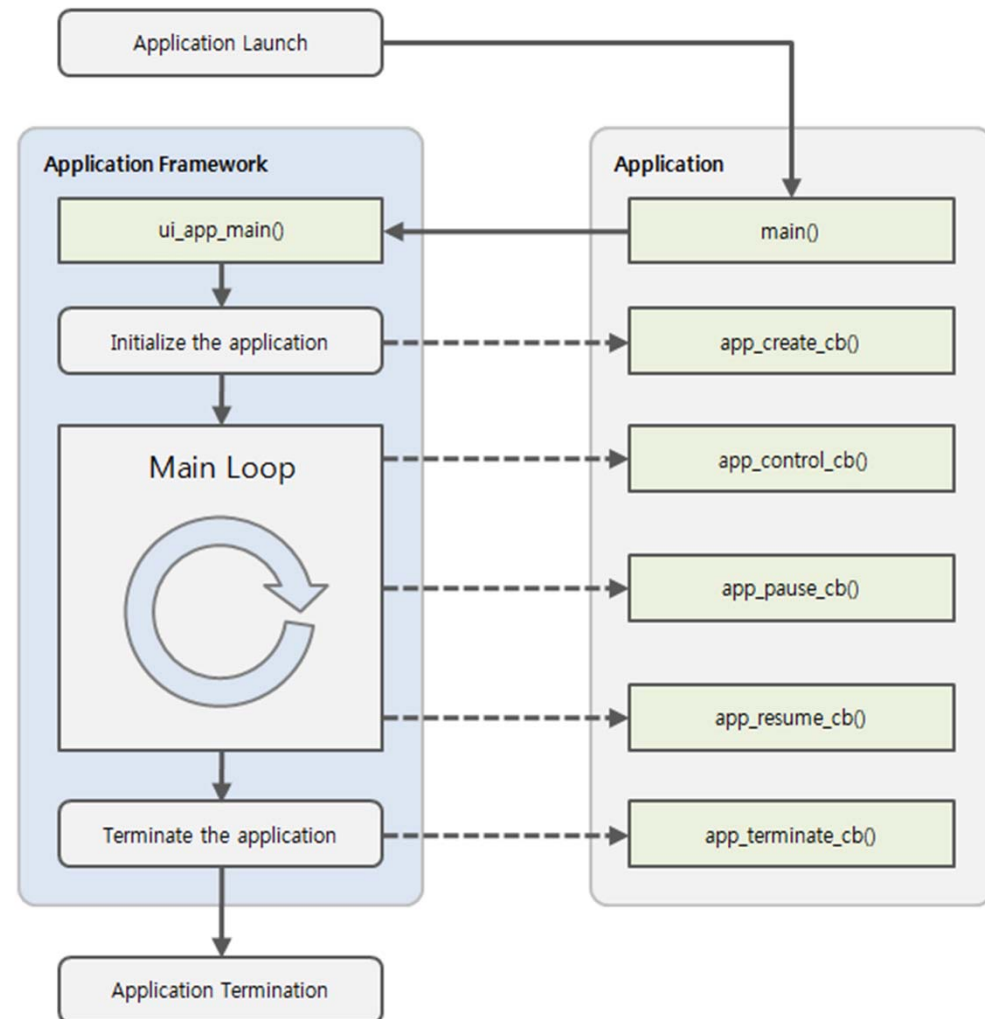


Figure1. UI and service application life-cycle

Appcore Callbacks



- ❑ The Application API defines 5 states with corresponding transition handlers



Application Example



❑ In the example native application

```
int
main(int argc, char *argv[])
{
    ui_app_lifecycle_callback_s event_callback = {0,};
    :
    :
    :
    event_callback.create = app_create;
    event_callback.terminate = app_terminate;
    event_callback.pause = app_pause;
    event_callback.resume = app_resume;
    event_callback.app_control = app_control;
    :
    :
    :
    ret = ui_app_main(argc, argv, &event_callback, &ad);

    return ret;
}
```

Application Example



```
int
main(int argc, char *argv[])
{
    appdata_s ad = {0,};
    int ret = 0;

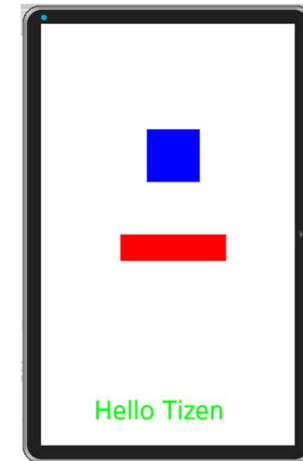
    ui_app_lifecycle_callback_s event_callback = {0,};
    app_event_handler_h handlers[5] = {NULL, };

    event_callback.create = app_create;
    event_callback.terminate = app_terminate;
    event_callback.pause = app_pause;
    event_callback.resume = app_resume;
    event_callback.app_control = app_control;

    ui_app_add_event_handler(&handlers[APP_EVENT_LOW_BATTERY], APP_EVENT_LOW_BATTERY, ui_app_low_battery, &ad);
    ui_app_add_event_handler(&handlers[APP_EVENT_LOW_MEMORY], APP_EVENT_LOW_MEMORY, ui_app_low_memory, &ad);
    ui_app_add_event_handler(&handlers[APP_EVENT_DEVICE_ORIENTATION_CHANGED], APP_EVENT_DEVICE_ORIENTATION_CHANGED, ui_app_orient_changed, &ad);
    ui_app_add_event_handler(&handlers[APP_EVENT_LANGUAGE_CHANGED], APP_EVENT_LANGUAGE_CHANGED, ui_app_lang_changed, &ad);
    ui_app_add_event_handler(&handlers[APP_EVENT_REGION_FORMAT_CHANGED], APP_EVENT_REGION_FORMAT_CHANGED, ui_app_region_changed, &ad);
    ui_app_remove_event_handler(handlers[APP_EVENT_LOW_MEMORY]);

    ret = ui_app_main(argc, argv, &event_callback, &ad);
    if (ret != APP_ERROR_NONE) {
        dlog_print(DLOG_ERROR, LOG_TAG, "app_main() is failed. err = %d", ret);
    }

    return ret;
}
```



Application Example



❑ In the example native application

```
int ui_app_main(int argc, char **argv, ui_app_lifecycle_callback_s *callback, void *user_data)
{
    struct appcore_ops appcore_context = {
        .data = &app_context,
        .create = _ui_app_appcore_create,
        .terminate = _ui_app_appcore_terminate,
        .pause = _ui_app_appcore_pause,
        .resume = _ui_app_appcore_resume,
        .reset = _ui_app_appcore_reset,
    };

    ○
    ○
    ○

    appcore_efl_main(app_context.app_name, &argc, &argv, &appcore_context);

    ○
    ○
    ○

    return APP_ERROR_NONE;
}
```

Application Example

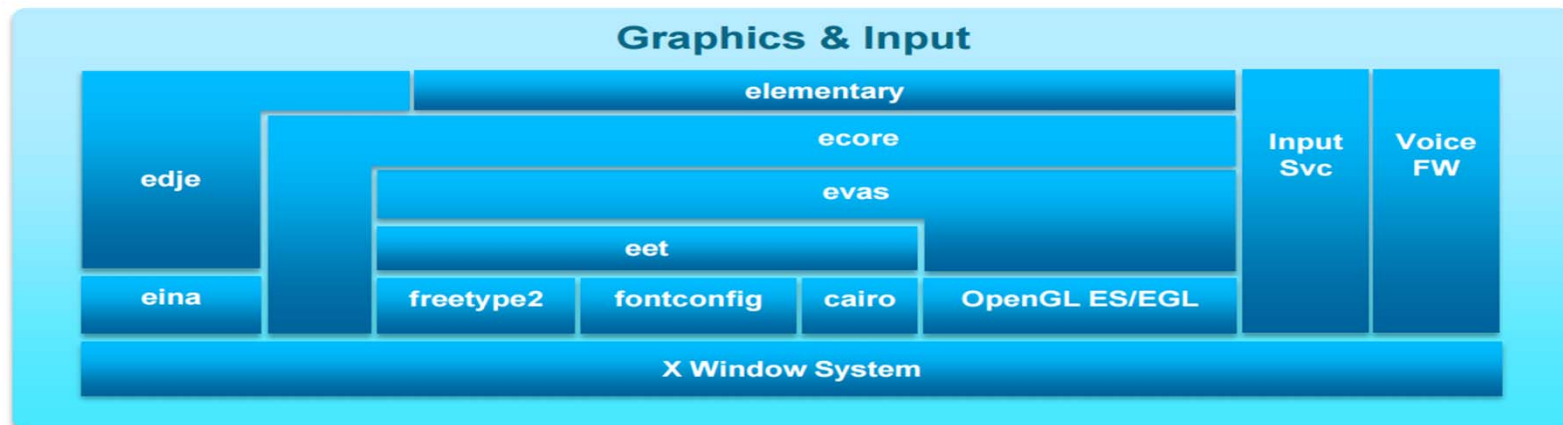
- ❑ EAPI void ecore_main_loop_begin (void)
 - This function will not return until @ref ecore_main_loop_quit is called.
 - **It will check for expired timers, idlers, file descriptors being watched by fd handlers**, etc.
 - Once everything is done, before entering again on idle state, any callback set as @c Idle_Enterer will be called.

Graphics & Input

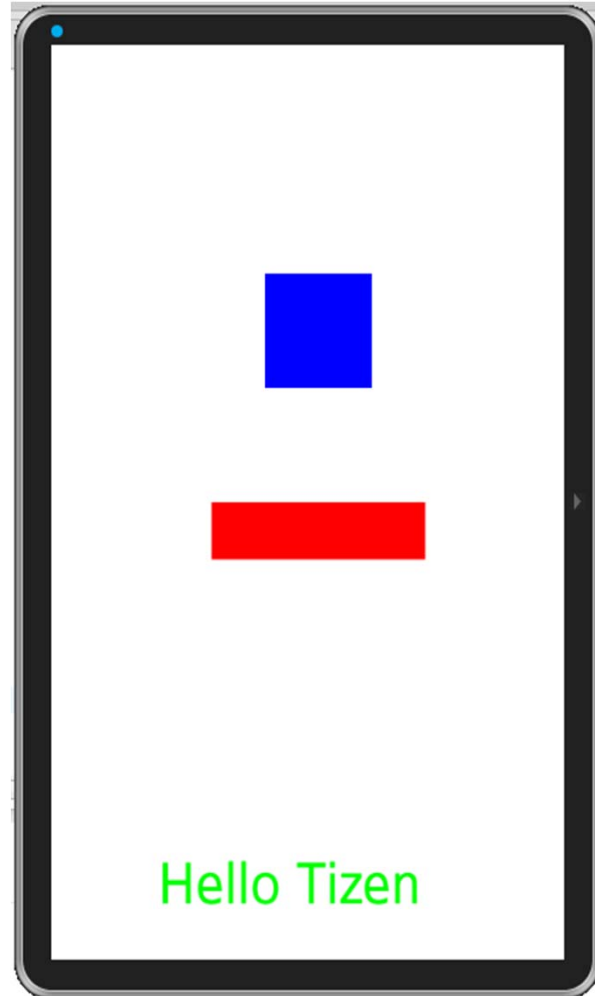


□ Consists of:

- Enlightenment Foundation Libraries
 - Rich Widgets multiple theme supports by Elementary
 - Retained mode canvas by Evas (Scene-graph, OpenGL ES back-end)
 - Compositing Window Manager
- Window System based on X11
- 3D (OpenGL ES), Font (freetype2, fontconfig)
- Input Service (SCIM), Voice FW (STT, TTS),



Application Example



```
static void
create_base_gui(appdata_s *ad)
{
    /* Window */
    ad->win = elm_win_util_standard_add(PACKAGE, PACKAGE);
    elm_win_autodel_set(ad->win, EINA_TRUE);

    if (elm_win_wm_rotation_supported_get(ad->win)) {
        int rots[4] = { 0, 90, 180, 270 };
        elm_win_wm_rotation_available_rotations_set(ad->win, (const int *)&rots, 4);
    }

    evas_object_smart_callback_add(ad->win, "delete,request", win_delete_request_cb, NULL);
    eext_object_event_callback_add(ad->win, EEXT_CALLBACK_BACK, win_back_cb, ad);

    Evas *e = evas_object_evas_get(ad->win);

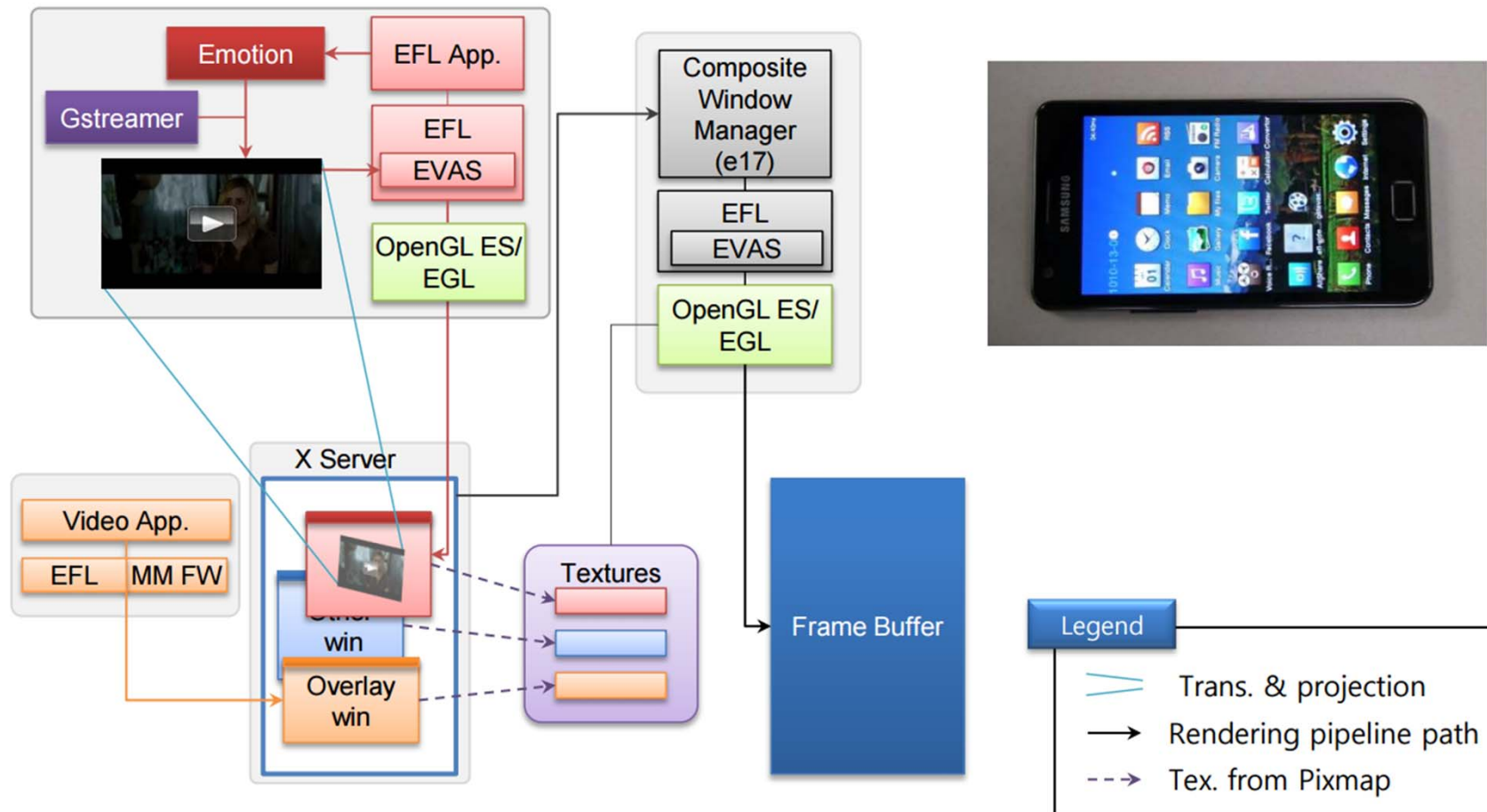
    Evas_Object *rec = evas_object_rectangle_add(e);
    evas_object_color_set(rec, 0, 0, 255, 255);
    evas_object_resize(rec, 100, 100);
    evas_object_move(rec, 200, 200);
    evas_object_show(rec);

    Evas_Object *rec2 = evas_object_rectangle_add(e);
    evas_object_color_set(rec2, 255, 0, 0, 255);
    evas_object_resize(rec2, 200, 50);
    evas_object_move(rec2, 150, 400);
    evas_object_show(rec2);

    Evas_Object *text;
    text = evas_object_text_add(e);
    evas_object_text_text_set(text, "Hello Tizen");
    evas_object_text_font_set(text, "DejaVu", 50);
    evas_object_color_set(text, 0, 255, 0, 255);
    evas_object_move(text, 100, 700);
    evas_object_show(text);
    evas_object_show(ad->win);
}
```

Graphics & Input: Advanced Feature

❑ Video decoding on an Evas object



Multimedia



❑ Provides:

- Playback of audio and video contents (local and streaming)
- Capturing images and recording audio and video
- 3D Audio Sound (OpenAL) specially for games
- Scanning & Playback of radio, Determining audio policy
- Extracting and displaying media content information

❑ Features:

- High Quality Video Playback
 - Full HD(1080P) Playback (with HW codec & Render Optimization)
 - Support for various kind of Multimedia Streaming (HTTP, RTP/RTSP)
 - Support for HTML5 Video and embedded playback in Web Browser
- High Quality & High Speed Camera/Recorder
 - High Quality Image Capture & Video Recording

Multimedia



❑ Key Components:

- GStreamer: Audio, Video, Recording, Streaming, Editing, Etc
- Audio Session Manager: Sound Policy Management
- PulseAudio: Software mixing multiple audio streams
- Multiple-Format Codec: Various support of codec
- Media Content Service: Content management for media files
- Audio I/O: Accessing raw audio buffer to manipulate



Web



❑ Provides:

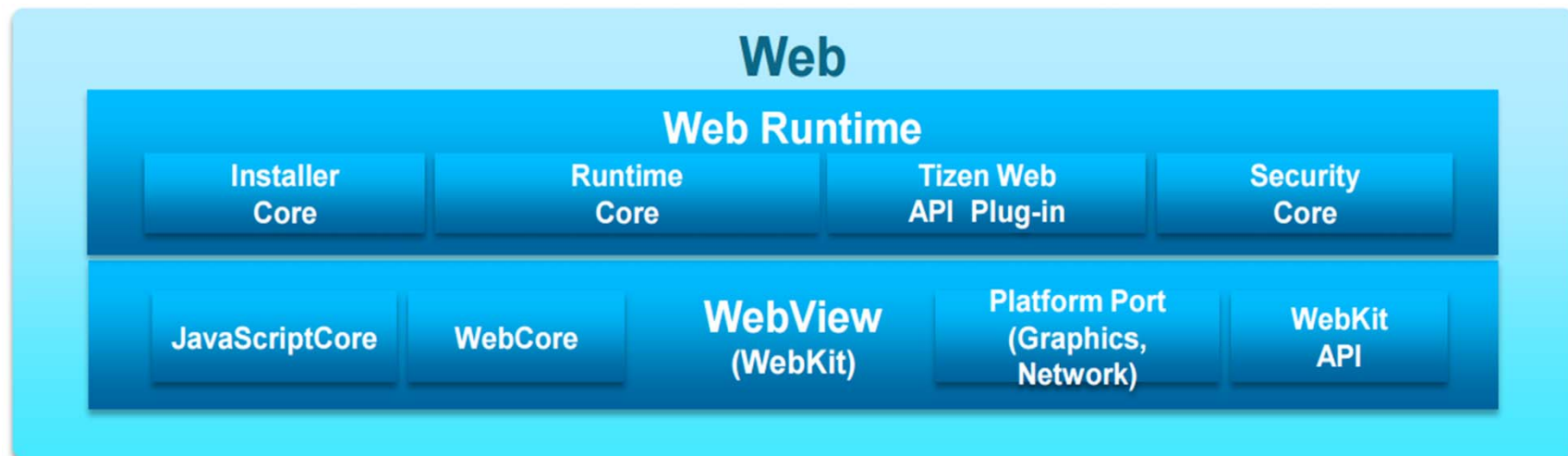
- Best Web experience with Browser and packaged Web Apps
 - Focusing on functionality(HTML5), performance (UI Responsiveness, 2D/3D)
 - Acceleration, JS Engine, Standard Compliance(W3C)
 - More device feature accessibility through Tizen Device API
 - jQuery Mobile based Tizen Web UI FW enables easy Web App development

Web



❑ Consists of:

- WebVeiw (WebKit2/EFL): JavaScriptCore, WebCore(HTML5/W3C API implementation), WebKit API
- Web Runtime: Execution environment for packaged Web Apps



Messaging



❑ Provides: SMS, MMS, Email

❑ Key Components

- Message Client API
- Message Server
 - Transaction Manager: Manage IPC between message server and library
 - Main Handlers: Handle message sending / receiving / filtering / setting
 - Storage Handlers: Save on DB
 - Plug-in Manager: Manage SMS and MMS Plug-ins



Location



❑ Provides

- Hybrid position information (GPS, SPS, WPS)
- Map Service (Geocode, POI, Route)

❑ Key Components

- GeoClue: Deliver location info from various positioning sources
- GeoClue library: An open source geo-information library
- GeoClue Providers: Implement the GeoClue library API
- Currently GPS Manager in GeoClue Providers is provided



Security

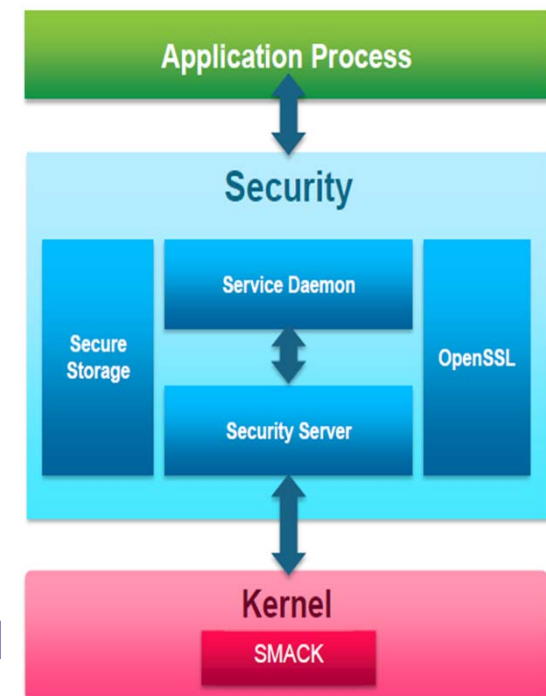


□ Provides

- Certificate management and verification
- Secure storage for confidential data
- User space access control management
- Cryptography and SSL support
- Mandatory access control support

□ Security model

- No root applications/No privilege escalation
- Sandboxed by SMACK(Simplified Mandatory Access Control Kernel)
- Service daemons will make use of SMACK and enforce access control in server side
- Manifest based permission policy for Apps



System



❑ System Manager

- Run as a daemon process
- Monitors device and system status and handles events from devices

❑ Sensor Manager

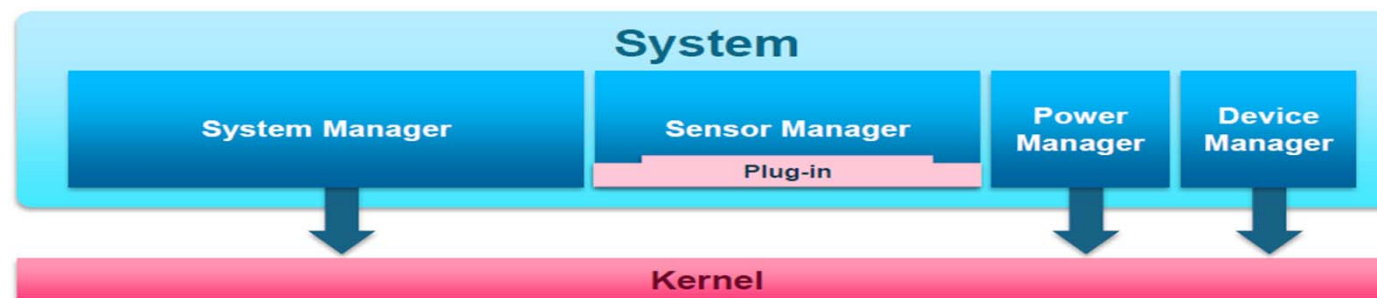
- Handling sensor events from various sensors

❑ Device Manager

- Setting/getting device values such as brightness

❑ Power Manager

- Controls LCD display backlight and application sleep



Base



- ❑ Base contains Linux base essential system libraries that provide key features.
- ❑ The Base is defined as self-sufficient and with packages in Base the system is able to boot itself to console/login.
- ❑ It also includes database support, internationalization, and XML parsing.

Connectivity



❑ Cellular and Wi-Fi Connection

- “Always-on” internet connections based on cellular(e.g.3G) and Wi-Fi .
- connman manages internet connections
 - Allowing automatic connection for available Wi-Fi access point
- Managing statistics of data network

❑ Bluetooth

- Based on Bluez and profiles (OPP, A2DP, RFCOMM, HFP, HDP, etc)
- Discovering / bonding / exchanging data with remote devices

Connectivity



❑ Tethering

- Providing three type of tethering : USB, Bluetooth and Wi-Fi

❑ NFC

- Including NFC Manager to handling NFC plug-ins
- Supporting P2P, Controlling NDEF tag, car emulator

❑ Wi-Fi

- Scanning and connecting Access Points
- Connecting hidden Access Points

Telephony



❑ Verified open source telephony stack

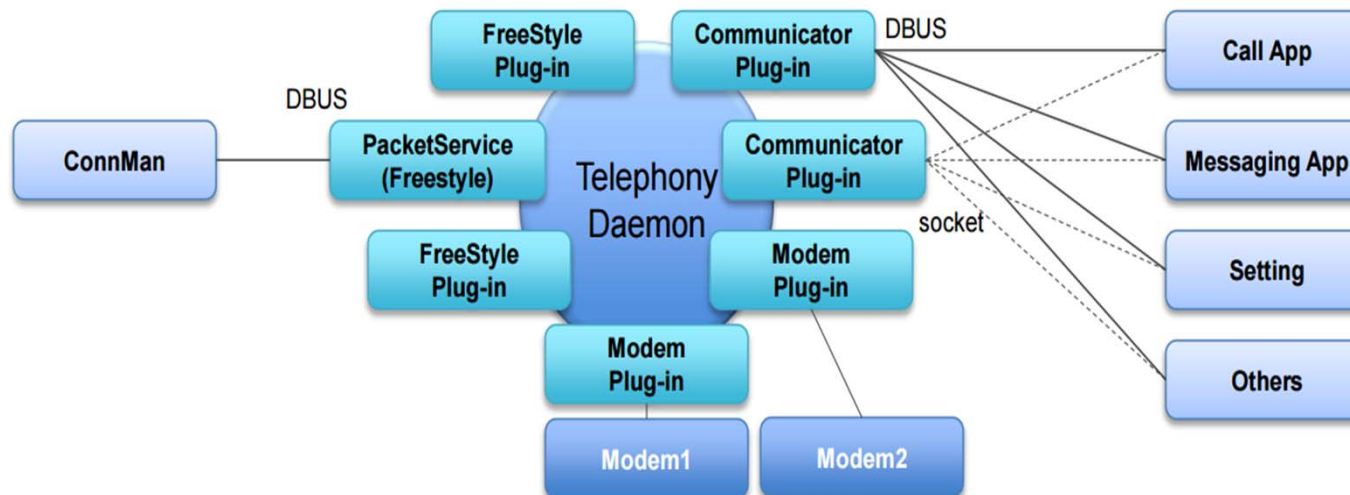
- It is a proven qualified stack with dominant industry modem chip vendors
- Applications in Tizen are already implemented on Tizen Telephony stack.
- It supports well-defined interface with ConnMan

Telephony



❑ Providing benefits for commercialization

- Flexible plug-in architecture for manufacturer's customization
- GCF, PTCRB-certified stack
- Manufacturer can make commercial product without license burden



*GCF : Global Certificate Forum

*PTCRB : PSC Type Certification Review Board

PIM



❑ Account

- Manage accounts to share account information on the device

❑ Contact/Calendar

- Account based, Multiple address/calendar books for an account.
- Enough features to satisfy mobile contact/calendar app requirements.

❑ Synchronization (Sync-FW)



CONCLUSIONS

Conclusions

- ❑ W3C standards-based with widest HTML5 coverage
- ❑ Targeting multiple device categories including smart phones, in-vehicle infotainment devices, smart TVs, computers, cameras, printers, and more
- ❑ Getting strong support from industry
- ❑ a Linux Foundation open source project based on Linux and various open source software

References

- ❑ <http://www.tizen.org> – main website
- ❑ <http://developer.tizen.org> – for application developer
- ❑ <http://source.tizen.org> – for platform developer
- ❑ <http://review.tizen.org/git> - source code
- ❑ <https://developer.tizen.org/forums> - tizen forms
- ❑ <http://wiki.tizen.org> – tizen wiki