Tizen Core APIs: A Core Framework Layer To Build In-House Applications

Jin-Woo Jeong
Tizen Architecture
Tizen Architecture

Web Framework
- HTML5/W3C
- Device APIs
- Web Runtime

Native Framework
- Social, Media, Web, Messages, Location
- UI/Graphics
- Base
- App/System/Security

Core
- Application Framework
- Security
- System
- Base
- Connectivity
- Telephony
- PIM

Kernel
- Linux Kernel & device drivers
Tizen Architecture | Frameworks

• **Public layers (Web & Native) focus on:**
  • Application development productivity
    • State-of-the-art HTML5/W3C APIs & Web UI framework
    • Full-featured native application development and features
  • Well-documented API references, developer guide, sample codes, and associated tools

• **Core sub-system focuses on:**
  • Providing common functionalities for Web and Native frameworks as an underlying layer
  • **Performance** optimization
Tizen Architecture | Application Types (1/2)

- **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 samples apps included in the SDK
Tizen Architecture | Application Types (2/2)

- **Core applications**
  - Apps using *internal APIs* to fully utilize device capabilities

- **In-house applications**
  - Pre-loaded Core applications developed by device implementers
  - Call app, Calculator app, Gallery app, Contacts app, etc.
Tizen Architecture | Core Framework & API Layer

In-house Applications & Services running smoothly on any Tizen mobile phones from low-end to high-end!
Core Applications
Tizen Architecture | Core Applications - Revisited

• What is Core Application?
  • Application written in Core APIs for Tizen
  • Fully utilizes device features

• User interface:
  • Enlightenment Foundation Libraries (EFL)

• Device features
  • App framework: application, package, etc.
  • Social: contacts, calendar, etc.
  • Multimedia: image, video, audio, etc.
  • Other device-related features
Tizen Architecture | Core Applications – App Types

• **UI application**
  • Applications with UI
  • Developed using EFL + Core APIs

• **Service application**
  • Applications without UI
  • Developed using Core APIs
**Application states**

<table>
<thead>
<tr>
<th>State</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>READY</td>
<td>The application is launched</td>
</tr>
<tr>
<td>CREATED</td>
<td>The application starts the main loop</td>
</tr>
<tr>
<td>PAUSED</td>
<td>The application is running but invisible to users</td>
</tr>
<tr>
<td>RUNNING</td>
<td>The application is running and visible to users</td>
</tr>
<tr>
<td>TERMINATED</td>
<td>The application is terminated</td>
</tr>
</tbody>
</table>

**State transition callbacks should be provided before starting the loop**
Tizen Architecture | Core Applications – App Lifecycle (2/4)

- Callbacks regarding App life-cycle

<table>
<thead>
<tr>
<th>Callback</th>
<th>Description</th>
<th>Action (Example)</th>
</tr>
</thead>
<tbody>
<tr>
<td>app_create_cb</td>
<td>Hook to take necessary actions before main event loop starts</td>
<td>UI generation code</td>
</tr>
<tr>
<td>app_pause_cb</td>
<td>Hook to take necessary actions when application becomes invisible</td>
<td>Releasing memory/resources</td>
</tr>
<tr>
<td>app_resume_cb</td>
<td>Hook to take necessary actions when application becomes visible</td>
<td>Re-allocating resources</td>
</tr>
<tr>
<td>app_terminate_cb</td>
<td>Hook to take necessary actions when your application is terminating</td>
<td>Release all resources</td>
</tr>
<tr>
<td>app_service_cb</td>
<td>Hook to take necessary actions for responding to the launch request</td>
<td>Required action</td>
</tr>
</tbody>
</table>
Tizen Architecture | Core Applications – App Lifecycle (3/4)

- Who can launch applications?
  - Users through the application launcher
  - Another application which needs to perform a specific operation (AppControl)

```c
#include "app.h"

void app_launch(void)
{
    struct app_service *app_service = app_service_create("app", "app");
    app_service_start(app_service);
    app_service_destroy(app_service);
}

app_launch();
```

http://tizen.org/appcontrol/operation/view
**Additional callbacks for system events**

<table>
<thead>
<tr>
<th>Callback</th>
<th>Description</th>
<th>Action (Example)</th>
</tr>
</thead>
<tbody>
<tr>
<td>app_low_memory_cb</td>
<td>Hook to take necessary actions in low memory situations</td>
<td>Save data into a persistent memory</td>
</tr>
<tr>
<td>app_low_battery_cb</td>
<td>Hook to take necessary actions in low battery situations</td>
<td>Stop heavy cpu/power consumption</td>
</tr>
<tr>
<td>app_device_orientation_cb</td>
<td>Hook to take necessary actions for handling a device orientation change</td>
<td>Change display orientation</td>
</tr>
<tr>
<td>app_language_changed_cb</td>
<td>Hook to take necessary actions for handling a language change event</td>
<td>Refresh the display with a new language</td>
</tr>
<tr>
<td>app_region_format_changed_cb</td>
<td>Hook to take necessary actions for handling a region change event</td>
<td>Update time to reflect the timezone change</td>
</tr>
</tbody>
</table>
Tizen Core APIs: Layout & Details
Core API | Definition

• Selected set of APIs in Core framework
  • Well-structured set to improve the usability of device features

• Not all the APIs in core framework are Core APIs
### Core API | Domain Layout (2/2)

<table>
<thead>
<tr>
<th>Namespace</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Framework</td>
<td>Application management</td>
</tr>
<tr>
<td>System &amp; Storage</td>
<td>System &amp; Device management</td>
</tr>
<tr>
<td>Network &amp; Web</td>
<td>Network &amp; Connectivity related-features</td>
</tr>
<tr>
<td>Telephony</td>
<td>Cellular functionalities</td>
</tr>
<tr>
<td>Media &amp; Content</td>
<td>Multimedia data management</td>
</tr>
<tr>
<td>Messaging</td>
<td>Messaging service</td>
</tr>
<tr>
<td>Social</td>
<td>Private information management</td>
</tr>
<tr>
<td>Location</td>
<td>Location-based service</td>
</tr>
<tr>
<td>UI &amp; UX</td>
<td>User interface and experience</td>
</tr>
<tr>
<td>Open-source Software</td>
<td>Linux-based essential system libraries</td>
</tr>
</tbody>
</table>
Core API | Application Framework

- Provides
  - Managing the main event loop of an application, managing application state changes, launching other applications using the package name, URI, or MIME type (Application)
  - Installing / Uninstalling applications (Package manager)
  - Information about applications (Application manager)
  - Managing notifications (Notification)
  - Passing messages between applications (Message-port)
  - “Add to home” feature (Shortcut)
Core API | System & Storage (1/2)

- Provides system and device management features
  - Interfaces for accessing devices such as sensors, USB, MMC, battery, CPU, and display (Sensor, System-server, Device)
  - Power management (Power)
  - Getting information about the device (System-info, Runtime-info)
  - System configuration, simple notification (Vconf)
  - Sending log output for debug activities (Dlog)

How long can I use my phone? Want to know my Tizen version!
Core API | System & Storage (2/2)

- Runtime-info example: Getting Wi-Fi status

```c
#include <runtime_info.h>
#include <log.h>

runtime_info_error_t ret;
int wifi_status = 0;

ret = runtime_info_get_value_int(RUNTIME_INFO_KEY_WIFI_STATUS, &wifi_status);
if (ret == RUNTIME_INFO_ERROR_NONE)
{
    switch(wifi_status)
    {
        case RUNTIME_INFO_WIFI_STATUS_DISABLED:
            LOGI("Wi-Fi is disabled.");
            break;
        case RUNTIME_INFO_WIFI_STATUS_UNCONNECTED:
            LOGI("Wi-Fi is enabled and network connection is not established.");
            break;
        case RUNTIME_INFO_WIFI_STATUS_CONNECTED:
            LOGI("Network connection is established in Wi-Fi network.");
            break;
        default:
            LOGI("Invalid status");
            break;
    }
}
else
{
    LOGE("runtime_info_get_value_int() failed");
}
```

**runtime_info_get_value_int**

- Key value to search (Wi-Fi hotspot enabled, tethering enabled, vibration enabled, and so on)
- You can get the current Wi-Fi status!
Core API | Location

- **Provides location-based services (LBS)**
  - Position information, satellite, GPS status (**Location-manager**)
  - Geocoding service (**Geocoder**)
    - Converting between geographical coordinates and textual address
  - Searching a point of interest, routes (**POI, Route**)

Want to find the slowest route to my wife!
Core API | Network & Web

- Provides network and connectivity related functionalities
  - Managing modem data connections (Connection)
  - Managing Bluetooth devices (Bluetooth)
  - Managing near field radio communication (NFC)
  - Managing Wi-Fi, USB, Bluetooth tethering services (Tethering)
  - Managing the serial communication (Serial)
  - Downloading the contents of a URL to the storage (Url-download)

Want to download TDC logo image file!
... using free Wi-Fi around me 😊
Core API | Telephony

- Provides cellular functionalities communicating with a modem
  - Managing call-related information and services (Call & TAPI)
    - Voice call & video call
  - Obtaining information from a SIM card (SIM)
    - Mobile country code (MCC), mobile network code (MNC), service provider name (SPN)
    - SIM card status
  - Accessing the cellular network status information (Network-info)
    - Roaming state, received signal strength indicator (RSSI), network type, service status, etc

Want to know my network type (GSM? UMTS? LTE?)
Core API | Messaging (1/2)

- **Provides messaging services**
  - SMS & MMS related services (**Msg-service**)
    - Creating, setting properties (recipients, body), and sending messages
    - Searching for messages
    - Registering callbacks for receiving notifications when new incoming messages found
  - Managing E-mails (**Email-service**)
    - Creating, setting properties (recipients, attachments, etc) of e-mail messages
    - Managing mailboxes, filtering rules
  - Push service (**Push**)

My girlfriend seems to automatically send this SMS: "Ok, I will be there soon"
Core API | Messaging (2/2)

- **SMS send example**

  1) Create a message

  ```
  static messages_message_t g_sms;
  MESSAGES_error_t e;
  /* Create a SMS message */
  e = messages_create_message(MESSAGES_TYPE_SMS, &g_sms);
  if (found_error(MESSAGES_ERROR_NONE, "messages_create_message", e))
    //handle error
  ```

  2) Set recipient's address

  ```
  char my_sms_buddies[] = "**********";
  e = messages_add_address(&g_sms, my_sms_buddies);
  if (e != MESSAGES_ERROR_NONE)
    LOGE("Failed to add recipient using messages_add_address().");
  return;
  ```

  3) Set Text for the message

  ```
  char content[] = "An Intruder has been detected in the cookie cupboard.";
  MESSAGES_error_t e;
  /* Create a text of the SMS */
  e = messages_set_text(&g_sms, content);
  if (found_error(MESSAGES_ERROR_NONE, "messages_set_text", e))
    return;
  ```

  4) Send an SMS

  ```
  e = messages_send_message(SVC, &g_sms);
  if (e != MESSAGES_ERROR_NONE)
    LOGE("Failed to send on sms message using messages_send_message().");
  ```
Core API | Media & Content

- Provides
  
  - Encoding, decoding, and transforming images (Image-util)
  
  - Transcoding a media file (Video-util)
  
  - Recording from the audio device and playing raw audio data (Audio-io, Sound-manager)
  
  - Playing multimedia contents from a file, network, and memory (Player)
  
  - Playing the tone and Waveform audio files (Tone-player, Wav-player)
  
  - Controlling a camera device (Camera)
  
  - Managing information about media files (Media-content)
Core API | Social

- **Provides PIM-related services**
  - Managing account information on the device *(Account)*
  - Managing calendar events and accessing calendar database *(Calendar)*
    - Insert, update, remove calendar records
    - Searching records with filters
  - Managing contacts and accessing contact database *(Contacts)*
    - Insert, update, remove contact & group records
    - Searching records with filters
    - Setting display options
    - SIM-related features

It’s safe to record wedding anniversary rather than my birthday
• Provides UI features & interaction services
  • Managing dynamic box (widget) service (Dynamic Box)
  • Playing simple sound and vibration (Feedback)
  • Synthesizing voice from text and playing synthesized sound data (TTS)
  • Recognizing the speech (STT)
  • Detecting and recognizing faces (Face)
Core API | Open Source Software

- Provides essential system libraries
  - Graphics
  - Window system
  - Internationalization
  - XML/JSON parsing
  - Database
  - Base utilities
  - …
Core API | API Style (1/2)

- **Naming convention**
  - Function: `{Namespace}_{Subject(optional)}_{Verb}_{Object}

```c
int camera_get_state(camera_h camera, camera_state_e *state);
int camera_set_capture_format(camera_h camera, camera_pixel_format_e format);
```

- Callbacks: typedef void (*namespace_actual_name_cb)([event_type event], [event details], void *user_data);

```c
typedef void (*app_service_cb) (service_h service, void *user_data);
typedef void (*camera_capture_completed_cb)(void *user_data);
```
Core API | API Style (2/2)

- **Data types**
  - Handles should have “_h” suffix
  - Enums should have “_e” suffix
  - Structures should have “_s” suffix

```c
int camera_get_state(camera_h camera, camera_state_e *state);
int camera_set_capture_format(camera_h camera, camera_pixel_format_e format);
```

- **Return type & values**
  - All core APIs should return an int type error value
  - Common error values are defined in tizen.h
  - Module-specific errors can be defined as enum
Core API | API Specification

• Core API headers are ready to generate API reference!

using Doxygen!

Module grouping

Brief introduction
Detailed information
Required parameters
Return values
Related APIs

TIZEN™ DEVELOPER CONFERENCE 2016 SAN FRANCISCO

33
How to Build/Install Core-Applications?

• Core application follows RPM packaging
  • GBS build

• Platform SDK supports building EFL applications
Q & A

• Thanks for listening!