Факультатив ФКН

РАЗРАБОТКА МОБИЛЬНЫХ ПРИЛОЖЕНИЙ НА БАЗЕ ПЛАТФОРМЫ

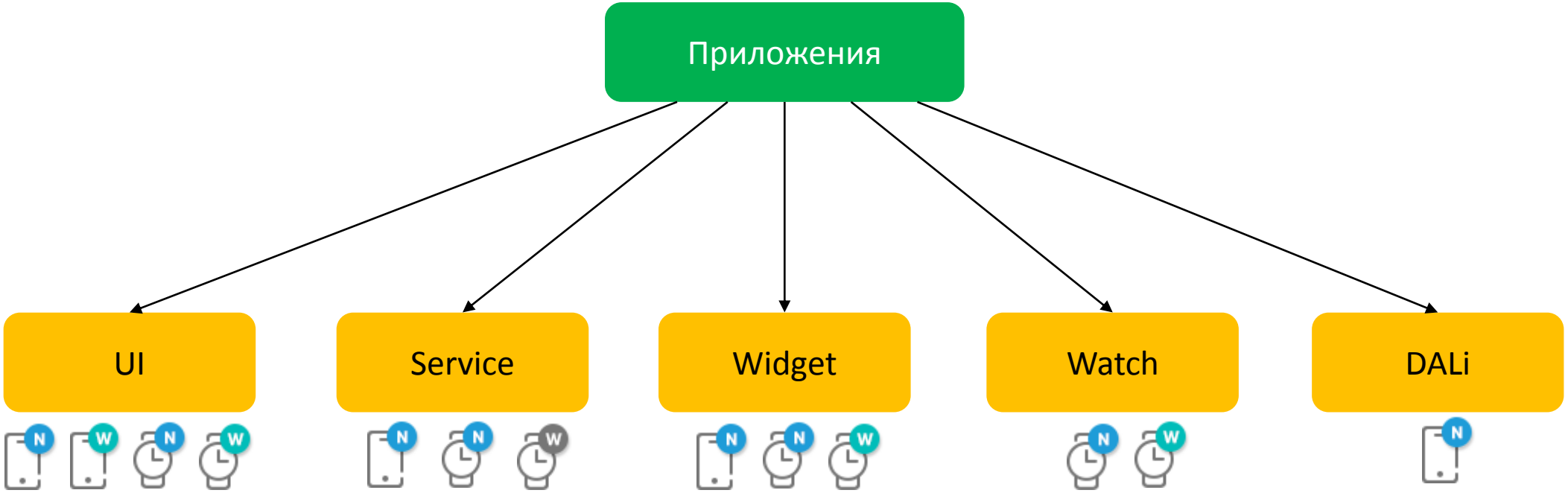
Преподаватель
Папулин Сергей Юрьевич (***papulin_hse@mail.ru***)

Ассистент
Цырлин Никита (***nstsyrlin@gmail.com***)

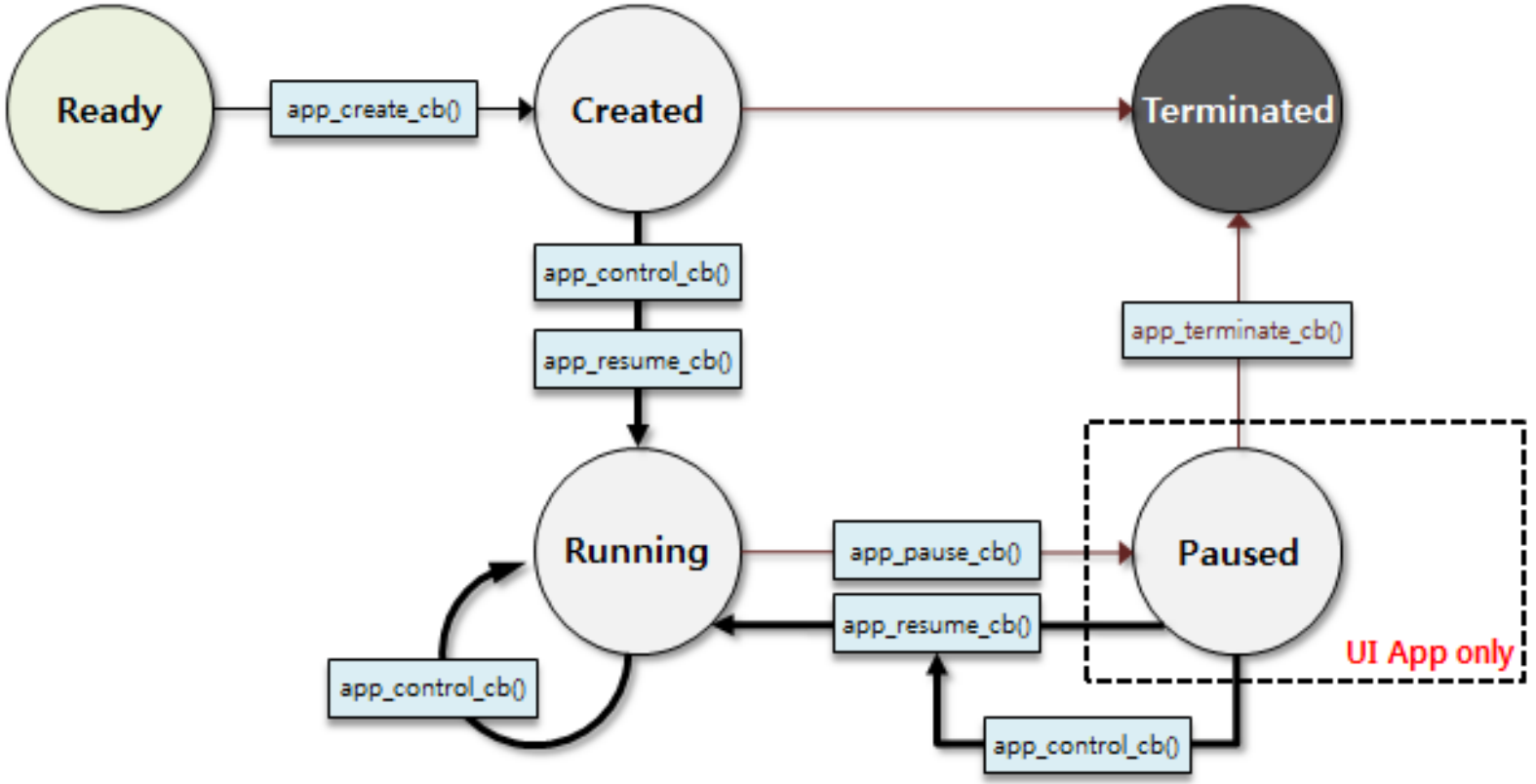
Москва, 2017

Нативные приложения

Приложения



Жизненный цикл приложения



https://developer.tizen.org/development/getting-started/native-application/tizen-application-model

Жизненный цикл приложения

`app_create_cb()` При запуске приложения выполняется перед запуском основного цикла (main loop)

`app_pause_cb()` При потере фокуса и переходе в состояние паузы (приложение невидимо):

- При запуске нового приложения по запросу и текущего приложения
- При переходе на домашнюю страницу (home screen)
- Системное событие (например, входящий вызов)
- При запуске приложения alarm’ом

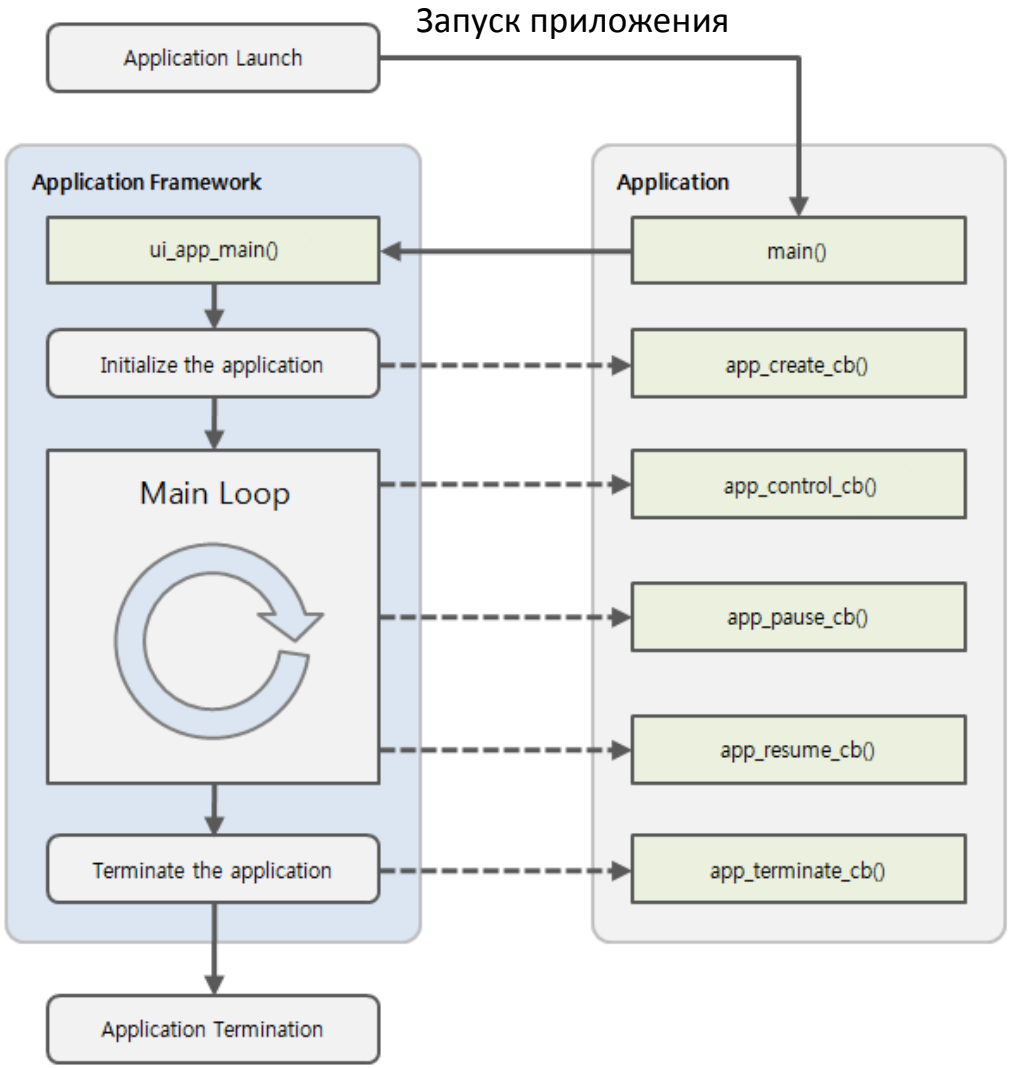
`app_resume_cb()` При возвращении приложения в активное состояние (видима пользователю)

- По запросу от другого приложения
- По завершению всех приложений поверх текущего в window-стеке
- При запросе от alarm’а

`app_terminate_cb()` При завершении приложения

- **`ui_app_exit()`** или **`service_app_exit()`**
- Нехватка памяти

Жизненный цикл приложения



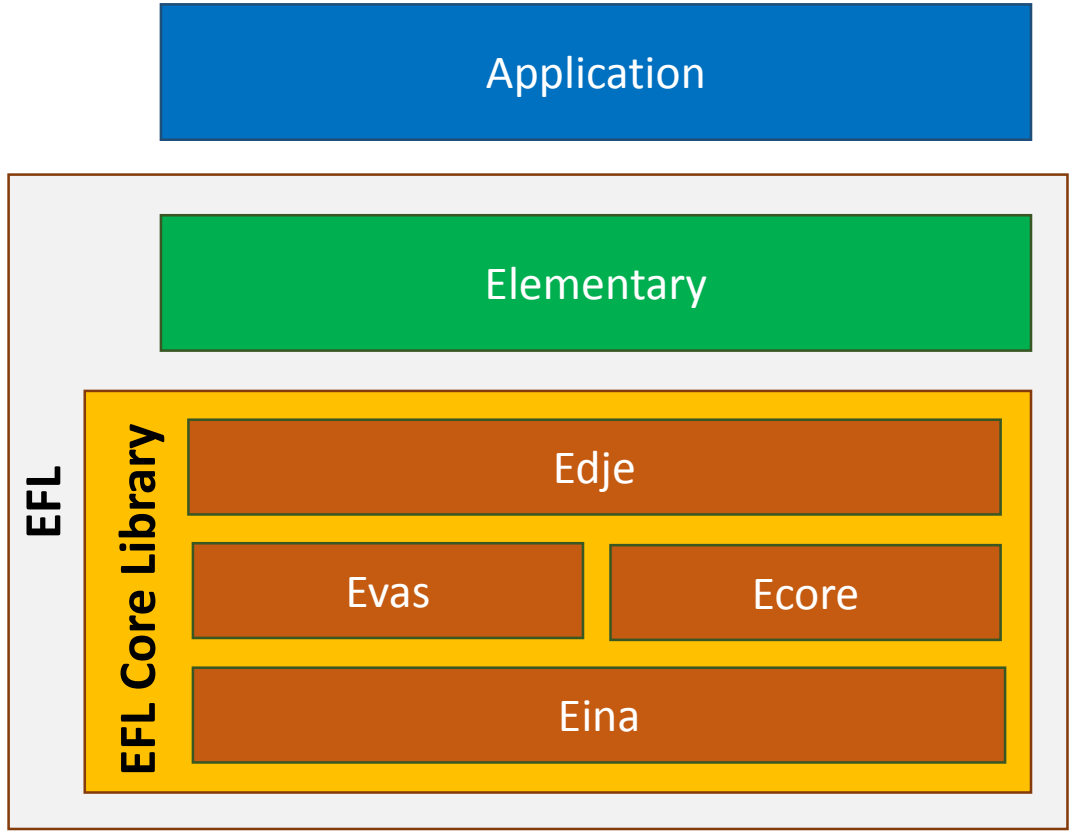
- Вызывается, когда запускается приложения
- Используется для инициализации пользовательского интерфейса
- Вызывается после app_create() при запуске приложения или когда приходит запрос на запуск при запущенном процессе приложения
- Вызывается, когда окно приложения полностью скрывается
- Вызывается, когда приложение возобновляет свою работы после паузы и появляется окно приложения
- Вызывается, когда приложение прекращает работу
- Вызывается, после выхода из основного цикла

Содержание директории нативного приложения

- `bin`: исполняемые файлы
- `lib`: библиотеки
- `res`: ресурсы
- `data`: данные приложения
- `shared/`: общие данные с другими приложениями

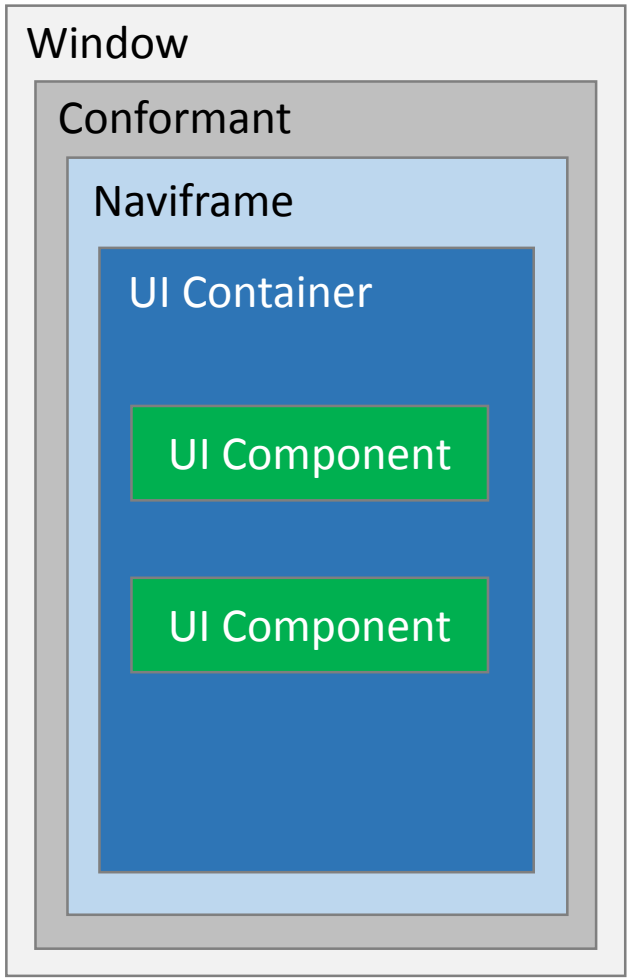
https://developer.tizen.org/development/getting-started/native-application/tizen-application-model

Пользовательский интерфейс с Enlightenment Foundation Libraries (EFL)



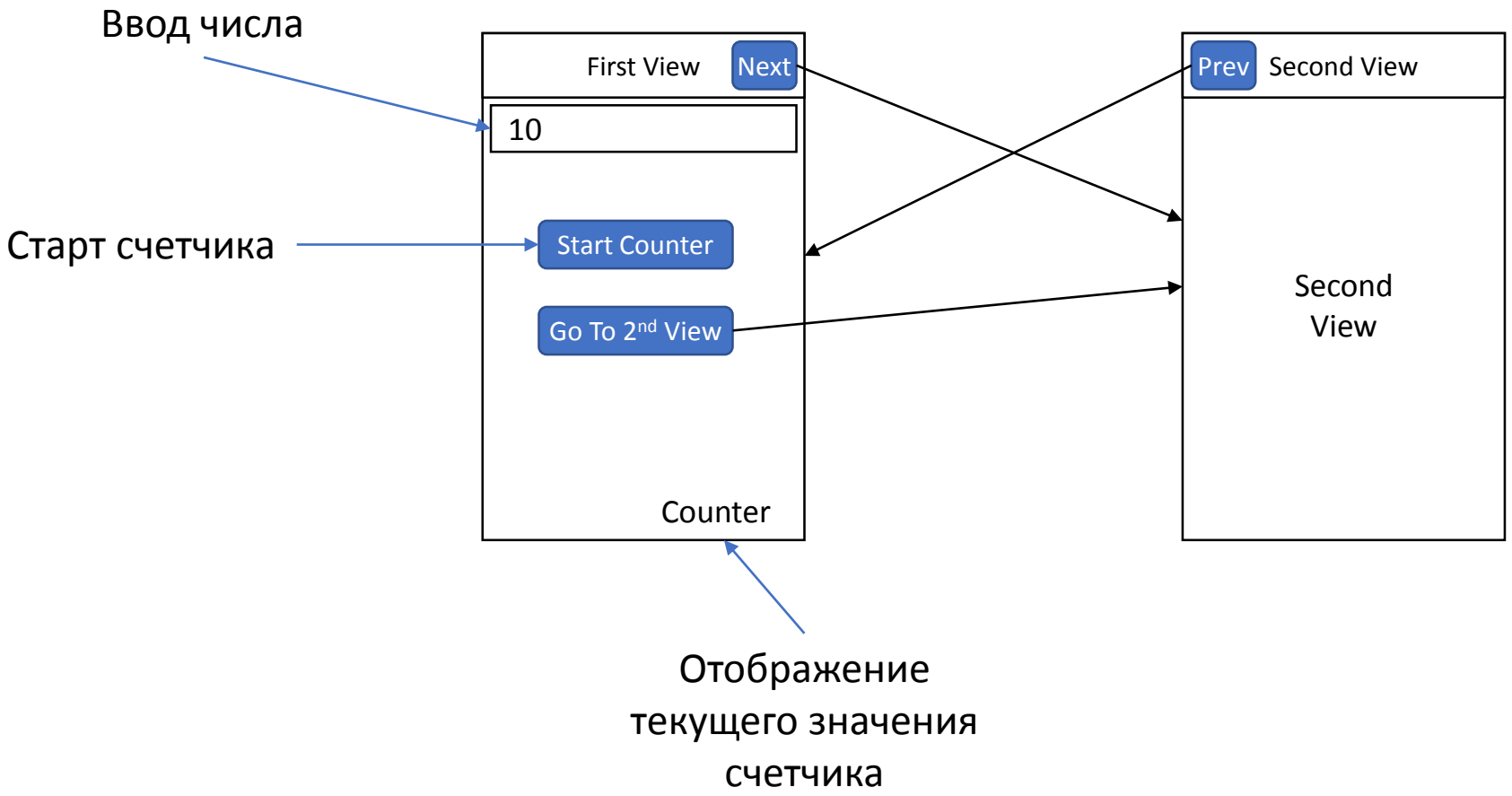
https://developer.tizen.org/development/guides/native-application/user-interface/efl/introduction

Размещение EFL UI

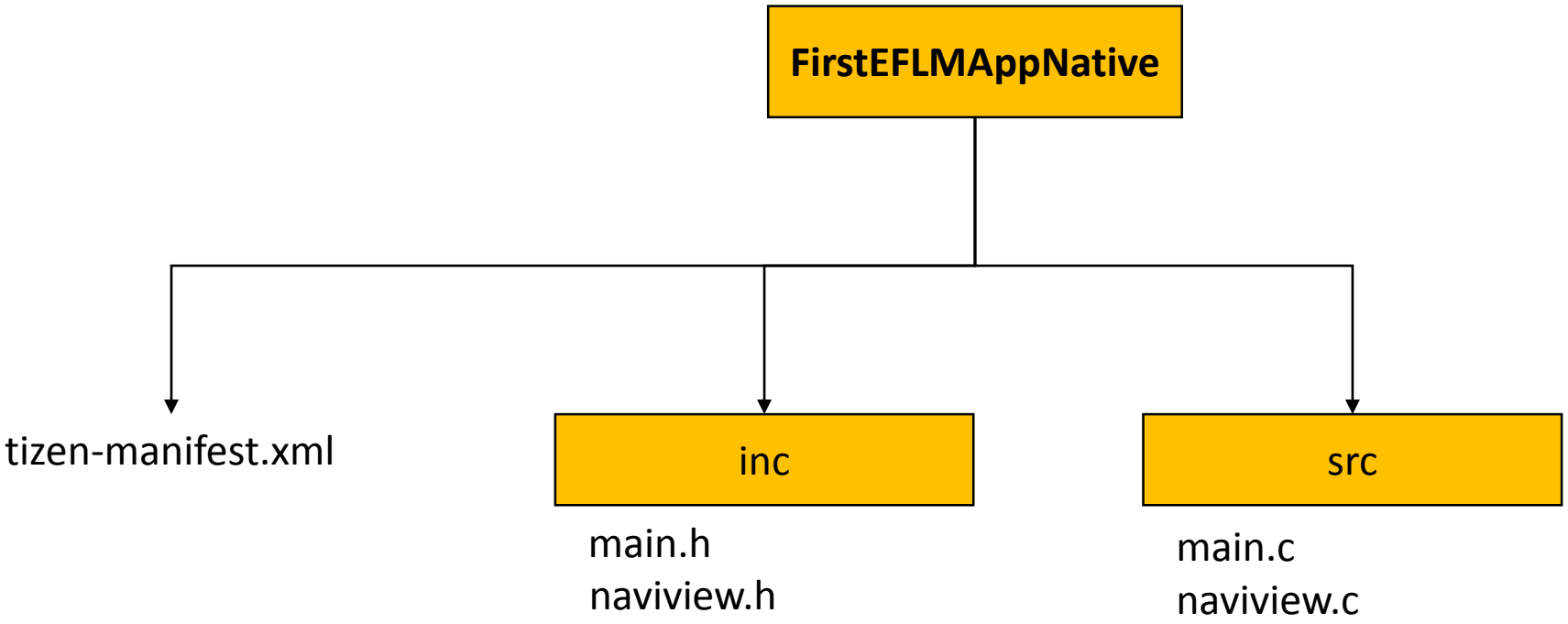


https://developer.tizen.org/development/guides/native-application/user-interface/efl/building-ui-layouts

Приложение **FirstEFLMAppNative**



Структура приложения



Заголовки

```
#define PACKAGE "org.example.firsteflmappnative"
```

main.h

```
typedef struct appdata {
    Evas_Object *win;
    Evas_Object *conform;
    Evas_Object *nf;
    Evas_Object *txt;
    Evas_Object *lbl;
} appdata_s;
```

naviview.h

```
void show_first_view_btn_clicked_cb(void *data, Evas_Object *obj, void *event_info);
void show_second_view_btn_clicked_cb(void *data, Evas_Object *obj, void *event_info);
void start_counter_btn_clicked_cb(void *data, Evas_Object *obj, void *event_info);
void create_first_view(appdata_s *ad);
void create_second_view(appdata_s *ad);
```

Создание первого View

```
// Создание формы first view
void create_first_view(appdata_s *ad)
{
    Evas_Object *content, *btn_nav, *btn_cont, *entry, *lbl;
    Elm_Object_Item *nf_it;

    // Создаем кнопку Next
    btn_nav = elm_button_add(ad->nf); // создаем кнопку для элемента naviframe
    elm_object_style_set(btn_nav, "naviframe/title_right"); // устанавливаем стиль кнопки
    elm_object_text_set(btn_nav, "Next"); // устанавливаем текст кнопки
    evas_object_smart_callback_add(btn_nav, "clicked", show_second_view_btn_clicked_cb, ad); //
    определяем функцию, которая будет выполнена при событии clicked

    // Box
    content = elm_box_add(ad->nf); // создаем контейнер типа box
    elm_box_align_set(content, 0, 0); // устанавливает его положение (верхний левый угол)
    evas_object_show(content);// устанавливаем свойство visible в true
    ...
```

Инициализация кнопки

```
// Кнопка в box'е
btn_cont = elm_button_add(ad->nf); // создаем кнопку в content для перехода на следующую страницу
elm_object_text_set(btn_cont, "Start Counter"); // устанавливаем текст
evas_object_size_hint_min_set(btn_cont, 400, 128); // устанавливаем размер
evas_object_size_hint_align_set(btn_cont, 0.5, 1); // определяем положение кнопки (внизу в центре)
evas_object_size_hint_weight_set(btn_cont, EVAS_HINT_EXPAND, 0.3); // размещение элемента в пространстве с
максимальной величиной по горизонтали  и на 0.5 по вертикали
evas_object_smart_callback_add(btn_cont, "clicked", start_counter_btn_clicked_cb, ad); // устанавливаем обработчик
события clicked
evas_object_show(btn_cont); // устанавливаем свойство кнопки visible=true
elm_box_pack_end(content, btn_cont); // добавляем кнопку в конец контейнера box
```

Переход между видами (views)

```
// Обработка события перехода на second view
void show_second_view_btn_clicked_cb(void *ad, Evas_Object *obj, void *event_info)
{
    // Создаем второй view
    create_second_view(ad);
}



// Обработка события перехода на first view
void show_first_view_btn_clicked_cb(void *data, Evas_Object *obj, void *event_info)
{
   Evas_Object *nf = data;

   // Извлекаем второй view из naviframe
   elm_naviframe_item_pop(nf);
}
```

Запуск счетчика

```
void start_counter_btn_clicked_cb(void *data, Evas_Object *obj, void *event_info) {

    appdata_s *ad = data;

    // в отдельном потоке
    ecore_thread_feedback_run(_long_function, _set_label_text, _end_func, NULL, ad, EINA_FALSE);
}
```

Ссылки

* ETL API:
https://developer.tizen.org/development/guides/native-application/user-interface/efl
https://developer.tizen.org/development/ui-practices/native-application/efl/ui-components
https://developer.tizen.org/development/ui-practices/native-application/efl/scaling/scalability-support
https://developer.tizen.org/development/ui-practices/native-application/efl/ui-containers/box