MEDIATEK

Launcher

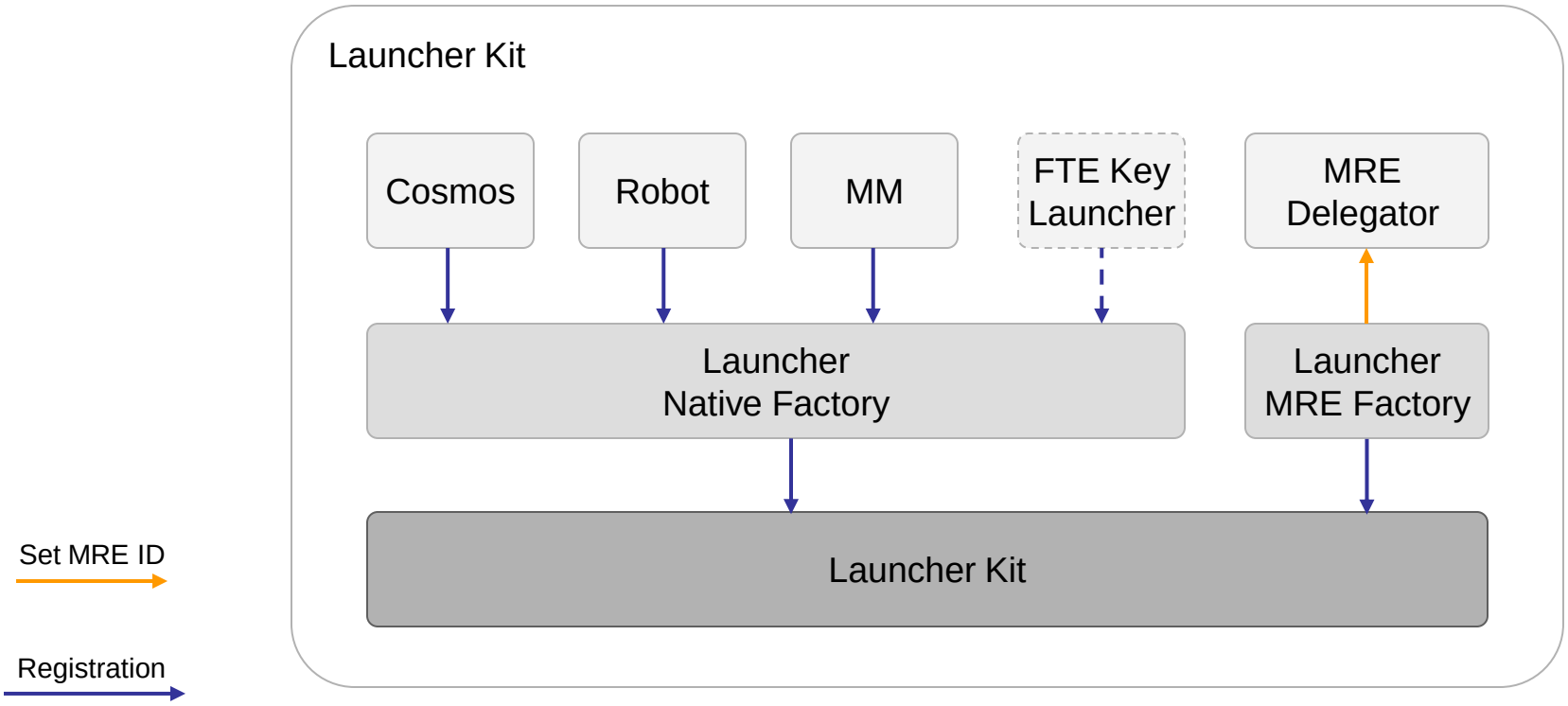




Outline

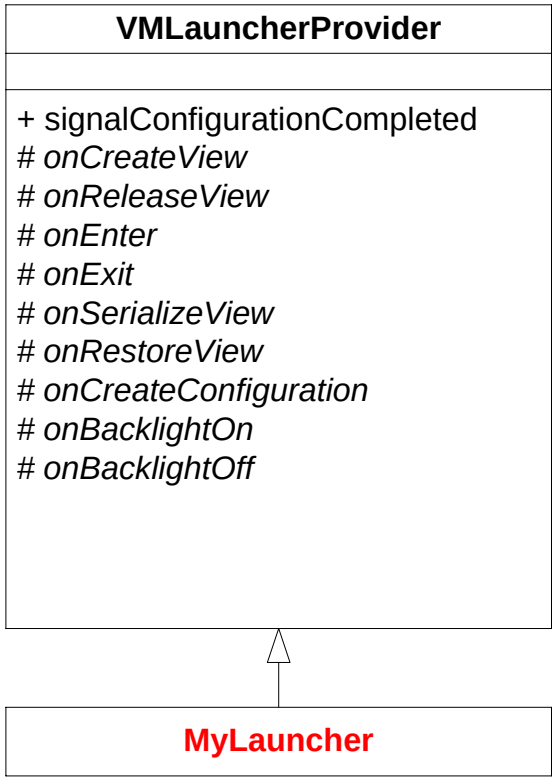
- Launcher Provider
  - Architecture
  - Class Diagram
  - Basic Class
- How Launcher Creates Widgets
  - Architecture
  - Class & Object Diagram
  - Basic Class
- How Launcher Creates Wallpaper
  - Architecture
  - Class & Object Diagram
  - Basic Class
- Hello World Explanation

Architecture: Launcher Kit



Launcher Provider Class Diagram

.\src\app\launcher**vm_launcher_provider.h**



Launcher Provider

- The base class of MRE launcher is ***VMLauncherProvider***
- Virtual Function:
  - **virtual void onCreateView()**
    - Notify app to create launcher's view
    - When the launcher is selected to put on the HS, *onCreateView* is called
    - When MS boots up and HS starts to load launcher, *onCreateView* is called
  - **virtual void onReleaseView()**
    - Notify app to release launcher's view
    - When the launcher is closed, *onReleaseView* is called
    - *To avoid from memory leak, we assert if any object isn't released after onReleaseView*

Launcher Provider

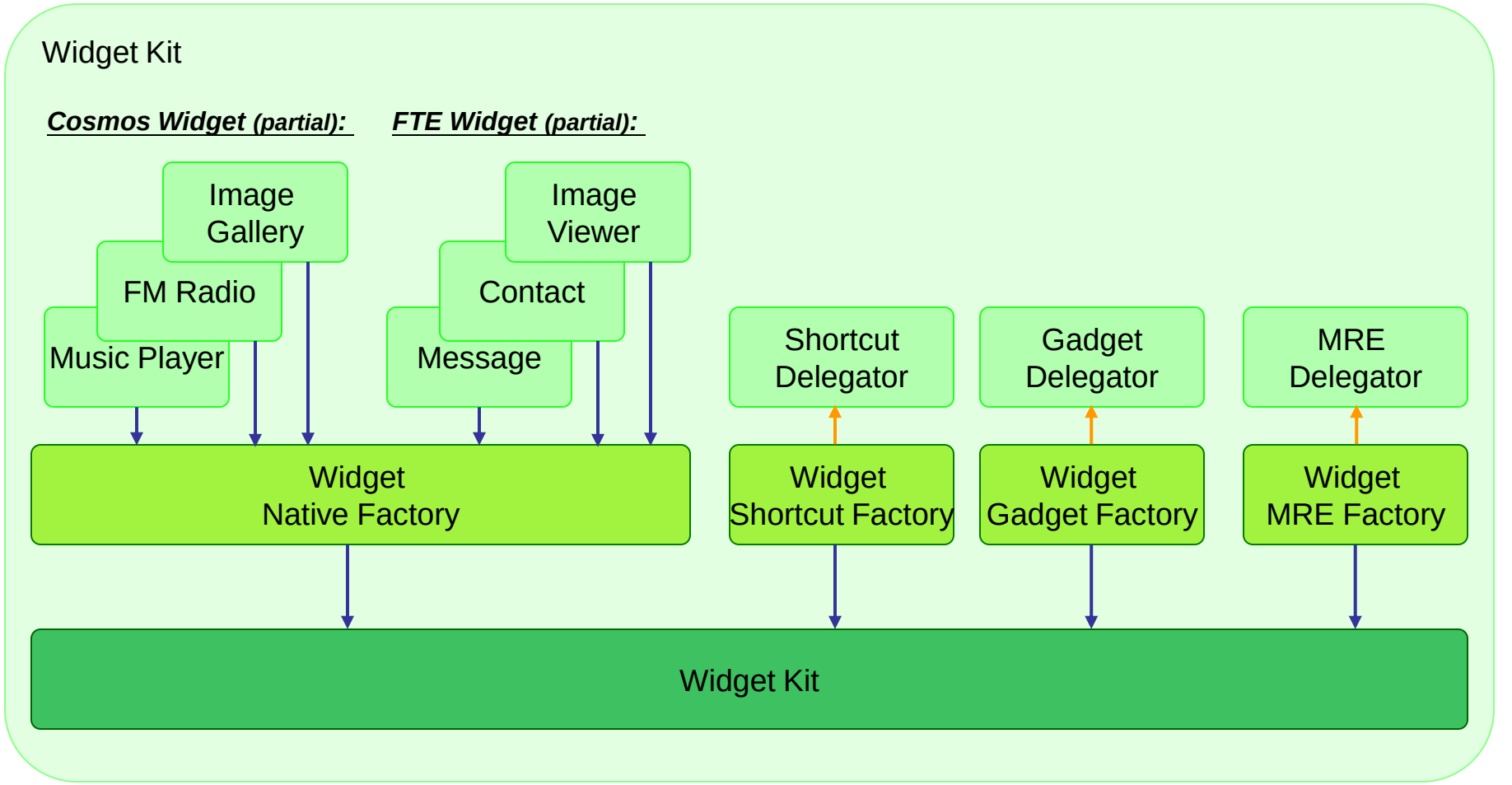
- **virtual void onEnter()**
  - Notify app that launcher's page is entered
  - E.g. Launcher can re-create the widget on the desktop
- **virtual void onExit()**
  - Notify app that launcher's page is exited
  - E.g. Launcher can close the widget on the desktop
- **virtual void onSerializeView()**
  - Notify app to serialize launcher's view when HS becomes inactive
  - Launcher can release parts of objects and memory
- **virtual void onRestoreView()**
  - Notify app to restore launcher's view when HS becomes active
  - Launcher can re-creates the object and memory

Launcher Provider

- **virtual void onCreateConfiguration()**
  - Notify app to create launcher's configuration

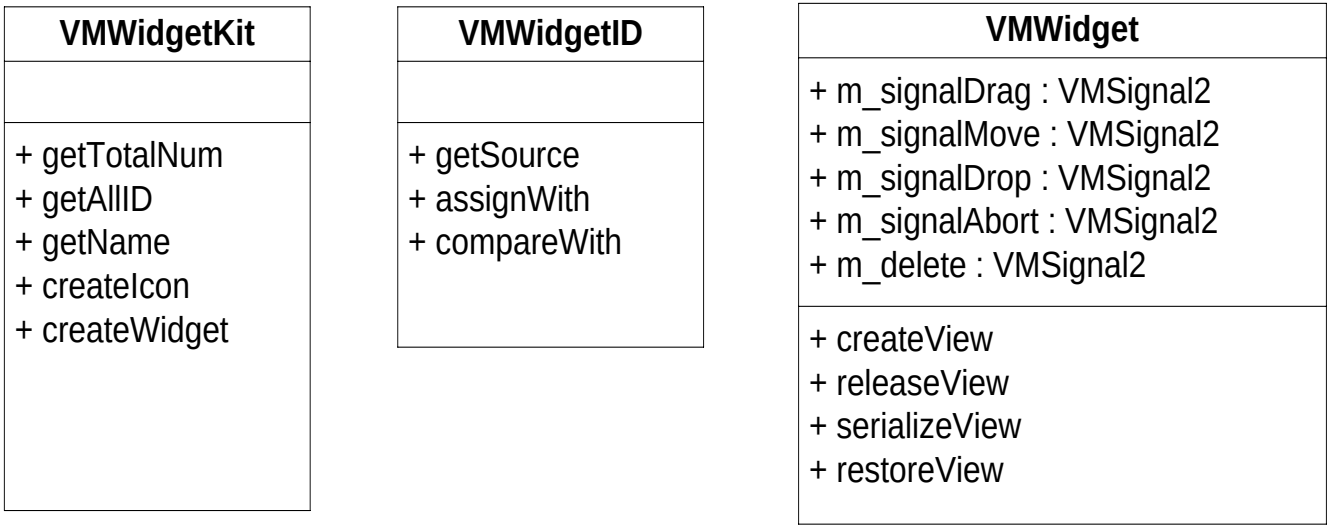
- Helper Function:
  - **void signalConfigurationCompleted(VMBOOL isSuccess);**
    - Call this function when configuration is completed or aborted

Architecture: Widget Kit

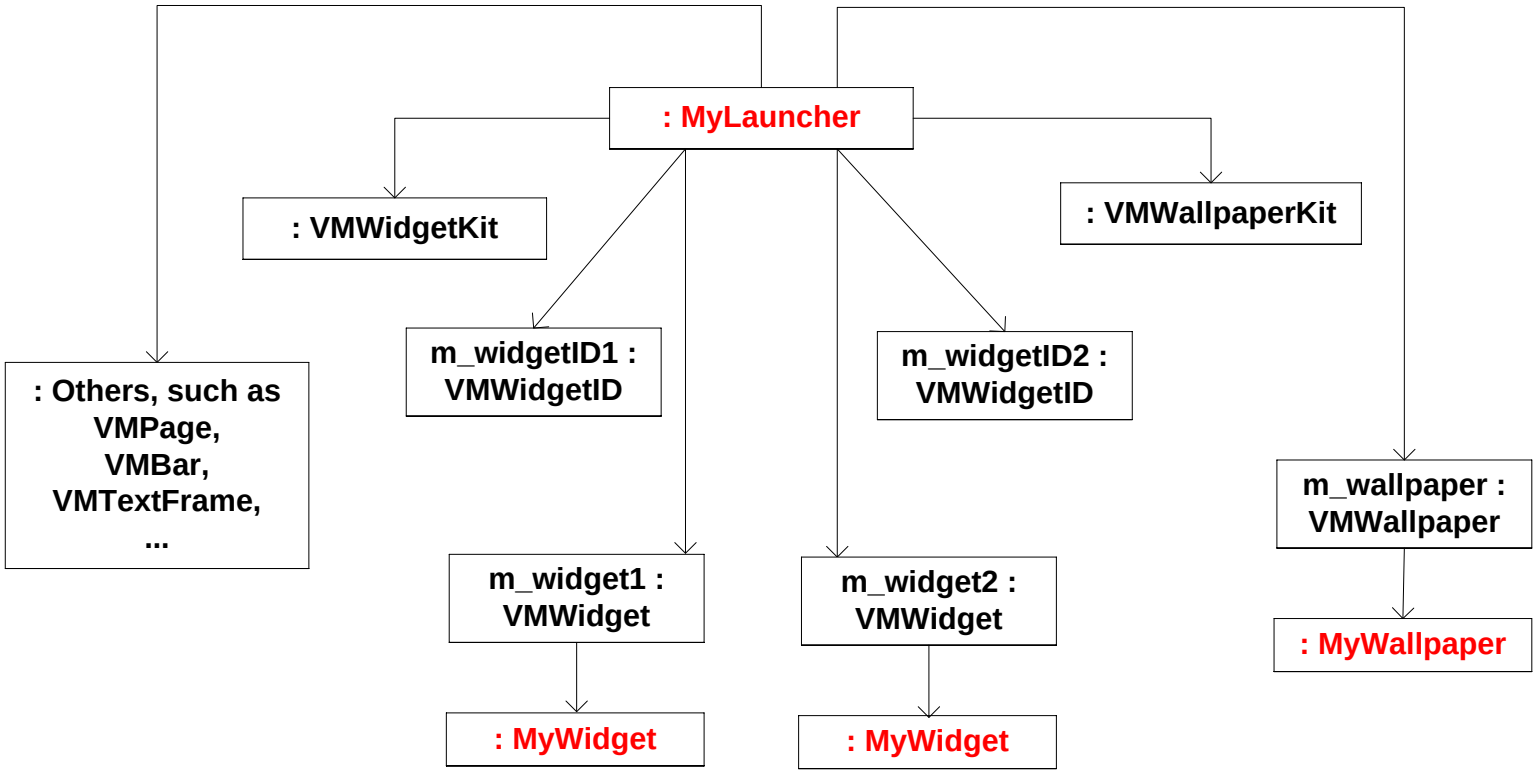


Widget Kit, Widget ID, Widget Class Diagram

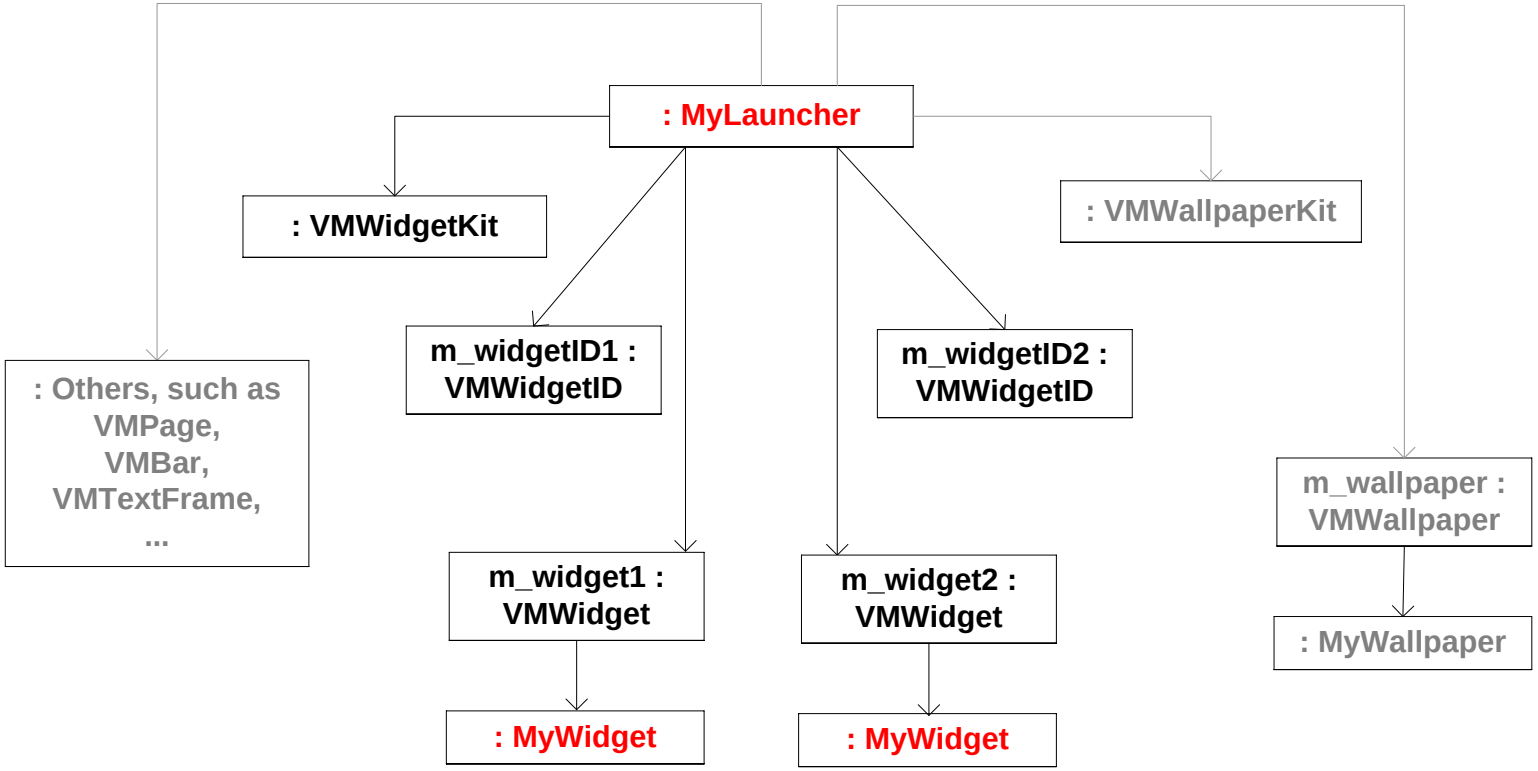
.\src\app\widget**vm_widget_kit.h**
.\src\app\widget**vm_widget_primitive.h**
.\src\app\widget**vm_widget.h**



Launcher Object Diagram



Widget Object Diagram



Widget Kit

- The class for MRE Launcher to create widget is ***VMWidgetKit***
- Help API
  - **VMUINT32 getTotalNum()**
    - Get Total number of supported widgets.
  - **VMUINT32 getAllId(**
    VMWidgetId **id, **// [OUT] Widget ID array**
    VMUINT32 size, **// [IN] Size of the array**
    VMBaseObject *parentObj) **// [IN] Parent of the ID object**
  - **VMUINT32 getAllId(**
    IVpiWidgetId **id,
    VMUINT32 size,
    IVpiObject *parentObj)
    - gets the ID of all the supported widgets.

Widget Kit

- **VMBOOL isIdValid(const IVpiWidgetId *id)**
  - Check if the ID is valid to create a widget

- **VMUINT32 getName(**
  const IVpiWidgetId *id, **// [IN] Widget ID**
  VMWCHAR *string, **// [OUT] String buffer**
  VMUINT32 size) **// [IN] Buffer size in wchar**
  - gets the name of the widgets

- **void createIcon(**
  IVpiFrame **icon, **// [OUT] Icon**
  const IVpiWidgetId *id, **// [IN] Widget ID**
  IVpiObject *parentObj) **// [IN] Parent of the ID object**
  - Create the widget icon

Widget Kit

- **VMWidget *createWidget(**
  const IVpiWidgetId *id, // [IN] Widget ID
  VMBaseObject *parentObj) // [IN] Parent Object
- **void createWidget(**
  IVpiWidget **widget, // [OUT] Widget
  const IVpiWidgetId *id, // [IN] Widget ID
  IVpiObject *parentObj) // [IN] Parent Object
  - Create widget

Widget ID

- The class for MRE Launcher to specify widget is ***VMWidgetID***
- Help API:
  - **VMWidgetSrcEnum getSource()**
    - Get source type of the widget

```
enum VMWidgetSrcEnum
{

   VM_WIDGET_SRC_UNKNOWN,
   VM_WIDGET_SRC_APP_SHORTCUT,     // Native app
   VM_WIDGET_SRC_SYSTEM_DEFAULT, // Native widget
   VM_WIDGET_SRC_DOWNLOAD,          // MRE widget or Google gadget
   VM_WIDGET_SRC_PROPRIETARY       // Embedded in this launcher
};
```

Widget ID

- **void assignWith(**
  const IVpiWidgetId *other)
  - Assign widget ID

- **VMBOOL compareWith(**
  const IVpiWidgetId *other) const
  - Compare if widget ID is the same

Widget

- The class for MRE Launcher to access widget is ***VMWidget***
- Help API:
  - **void createView()**
    - Create the widget's view
  - **void releaseView()**
    - Release the widget's view
  - **void serializeView()**
    - Serialize the widget's view when HS becomes inactive
    - Widget can release parts of objects and memory
  - **void restoreView()**
    - Restore the widget's view when HS becomes active
    - Widget can re-creates the object and memory

Widget

- Signal need to be handled:
  - **VMSignal2 <**

    VMWidget *, **// [IN] This widget**

    const vm_pen_event_struct & **// [IN] Pen event**

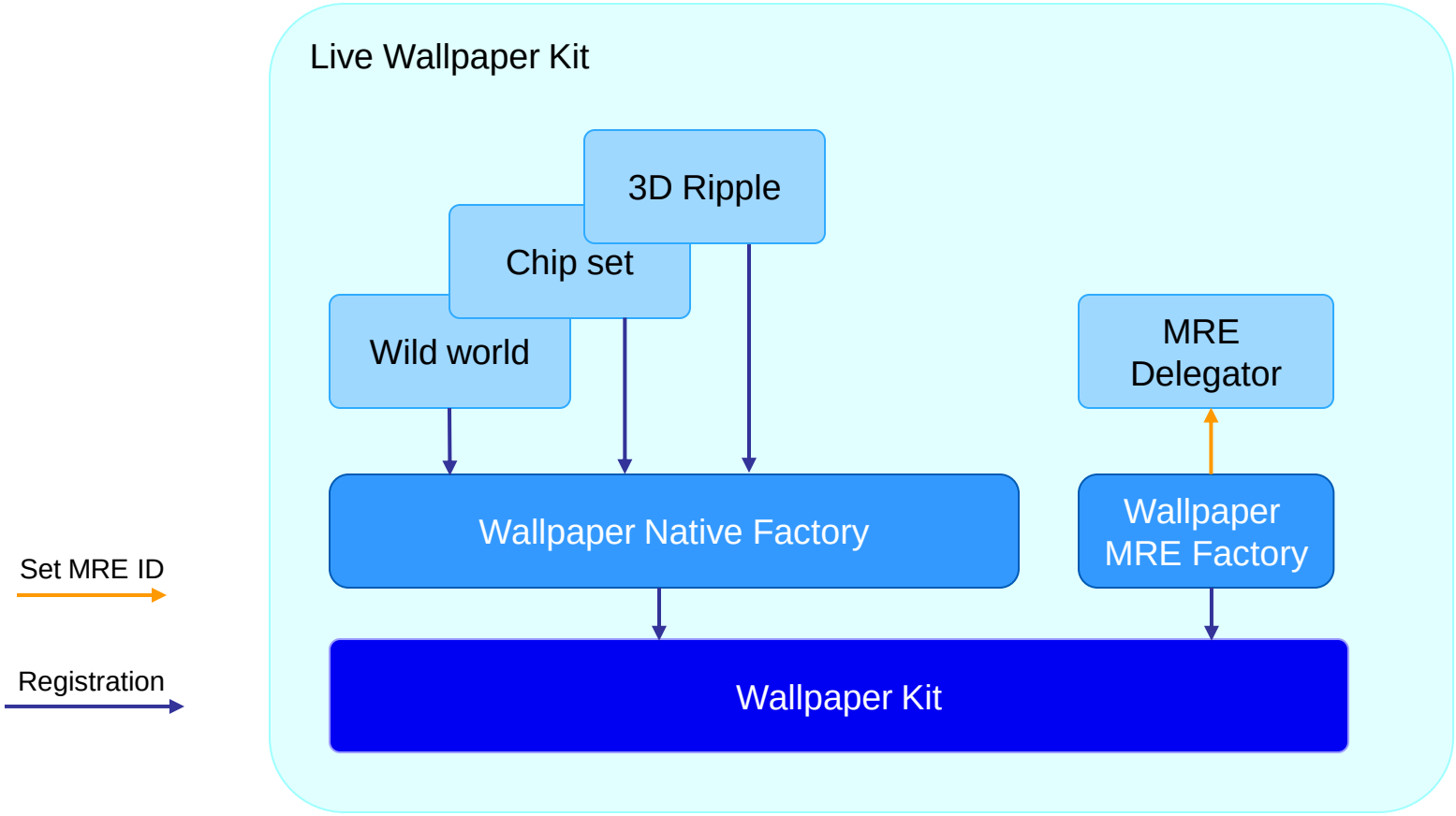
    > m_signalDrag, m_signalMove, m_signalDrop, m_signalAbort
    - This signal is emitted when the widget wants to be dragged, moved, dropped and abort the dragging.

  - **VMSignal1 <**

    VMWidget * **// [IN] This widget**

    > m_signalDelete;
    - This signal is emitted when the widget wants to be deselected from the home screen

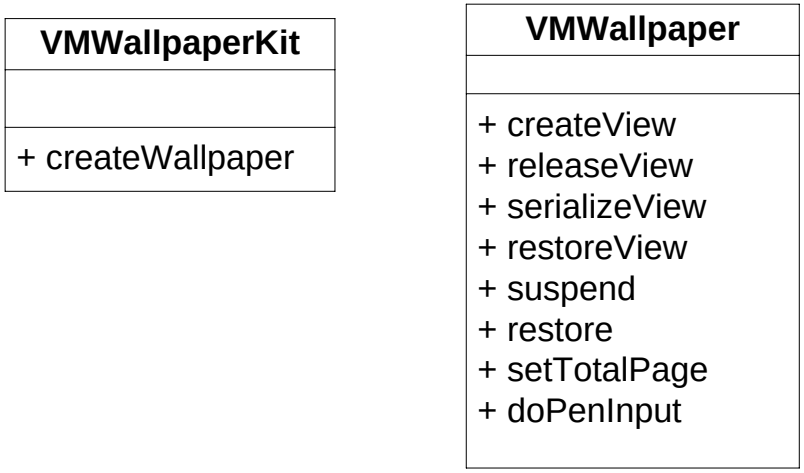
Architecture: Wallpaper Kit



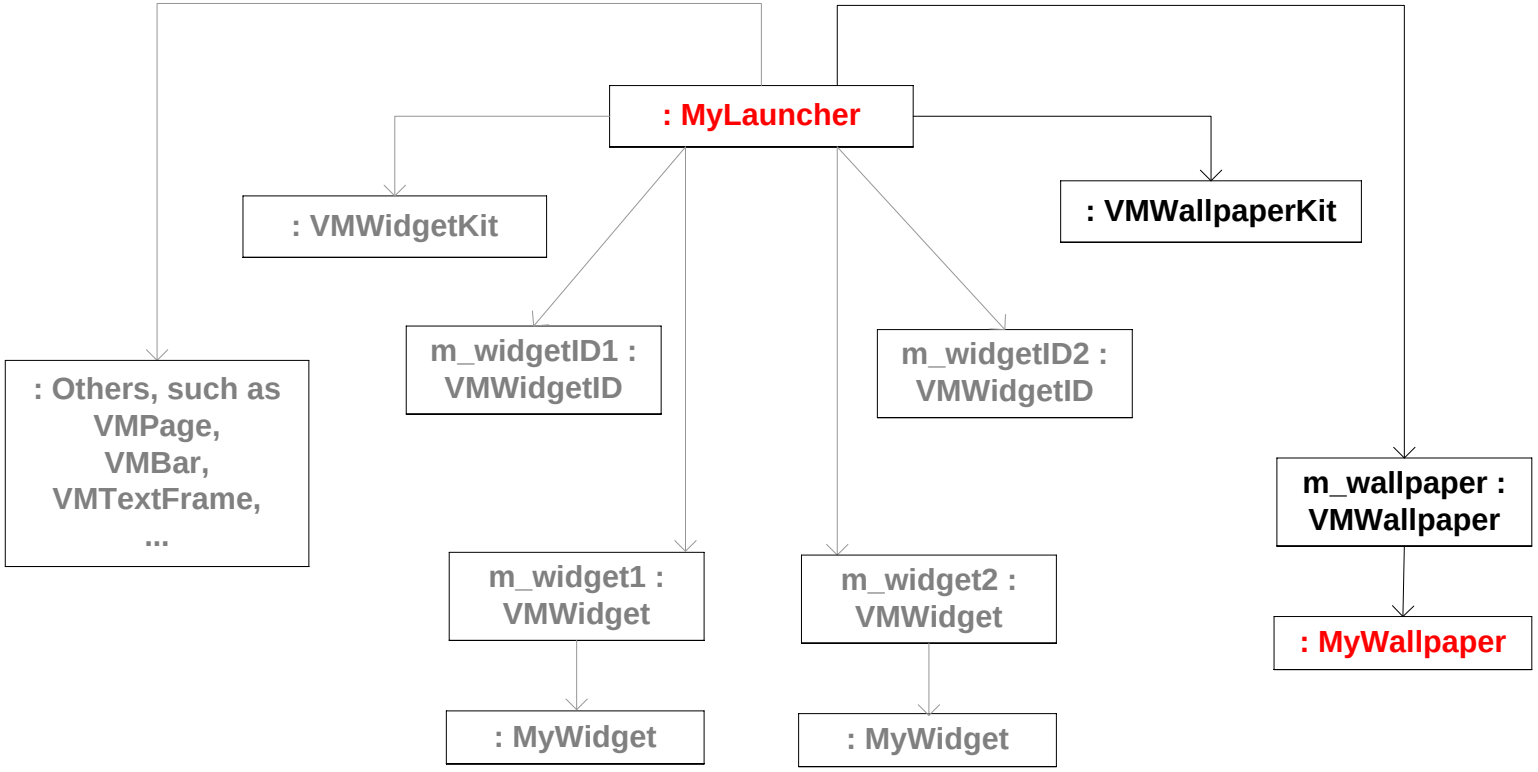
Wallpaper Kit & Wallpaper Class Diagram

.\src\app\wallpaper**vm_wallpaper_kit.h**

.\src\app\wallpaper**vm_wallpaper.h**



Wallpaper Kit Object Diagram



Wallpaper Kit

- The class for MRE Launcher to create wallpaper is ***VMWallpaperKit***
- Help API
  - **void createWallpaper(**
    IVpiWallpaper **wallpaper, **// [OUT] Wallpaper**
    IVpiObject *parentObj, **// [IN] Parent object of the wallpaper**
    vm_wallpaper_src_enum src, **// [IN] Wallpaper source**
    VMINT32 desktopCount = 1) **// [In] Destop count**
  - **VMWallpaper *createWallpaper(**
    VMBaseObject *parentObj, **// [IN] Parent object of the wallpaper**
    vm_wallpaper_src_enum src, **// [IN] Wallpaper source**
    VMINT32 desktopCount = 1) **// [In] Destop count**
    - Create wallpaper and it will call createView() automatically

Wallpaper

- The class for MRE Launcher to access wallpaper is ***VMWallpaper***
- Help API:
  - **void createView()**
    - Create the wallpaper's view
  - **void releaseView()**
    - Release the wallpaper's view
  - **void serializeView()**
    - Serialize the wallpaper's view when HS becomes inactive
    - Wallpaper can release parts of objects and memory
  - **void restoreView()**
    - Restore the wallpaper's view when HS becomes active
    - Wallpaper can re-creates the object and memory

Wallpaper

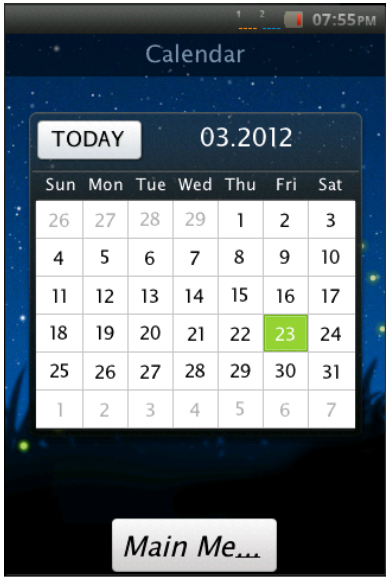
- **void suspend()**
  - Suspend the wallpaper
- **void resume()**
  - Resume the wallpaper
- **void setTotalPage(VMINT32 page)**
  - Set the total page number of wallpaper
  - E.g. Home Screen Wallpaper has page number greater than 1
  - E.g. Screen Lock Wallpaper has single page
- **void doPenInput(vm_pen_event_struct &event)**
  - Send key event to wallpaper
  - Because wallpaper almost can’t receive pen event from Venus framework

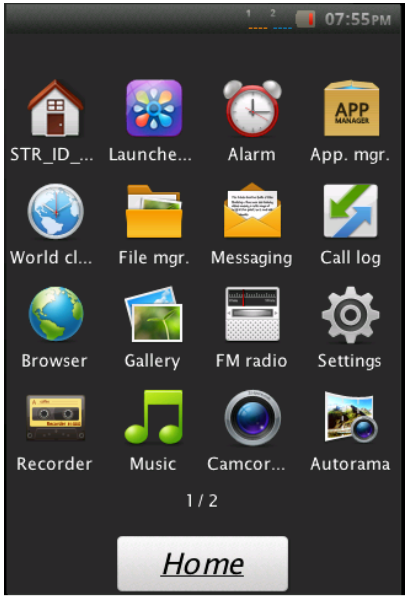
Launcher Provider – Hello World

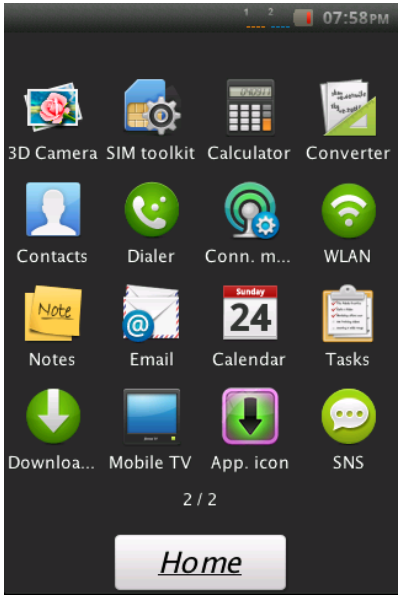
- Hello World Explanation

Click title to create the next widget

Swipe to change page





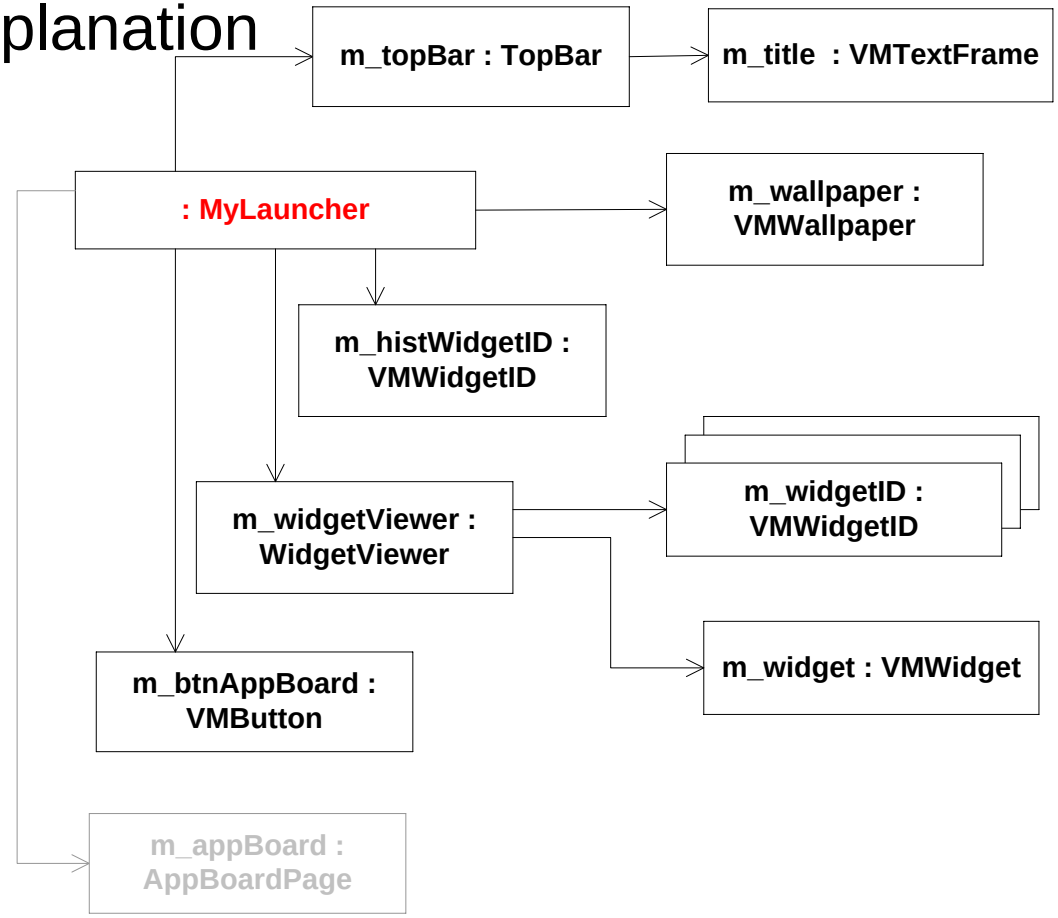


Click 'Main Menu' to push main menu page

Click 'Home' to go back to home

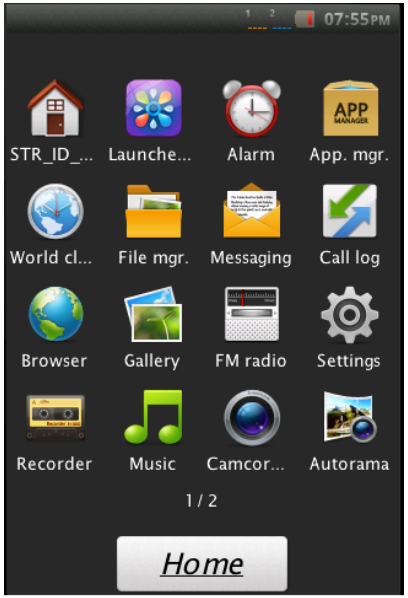
Launcher Provider – Hello World

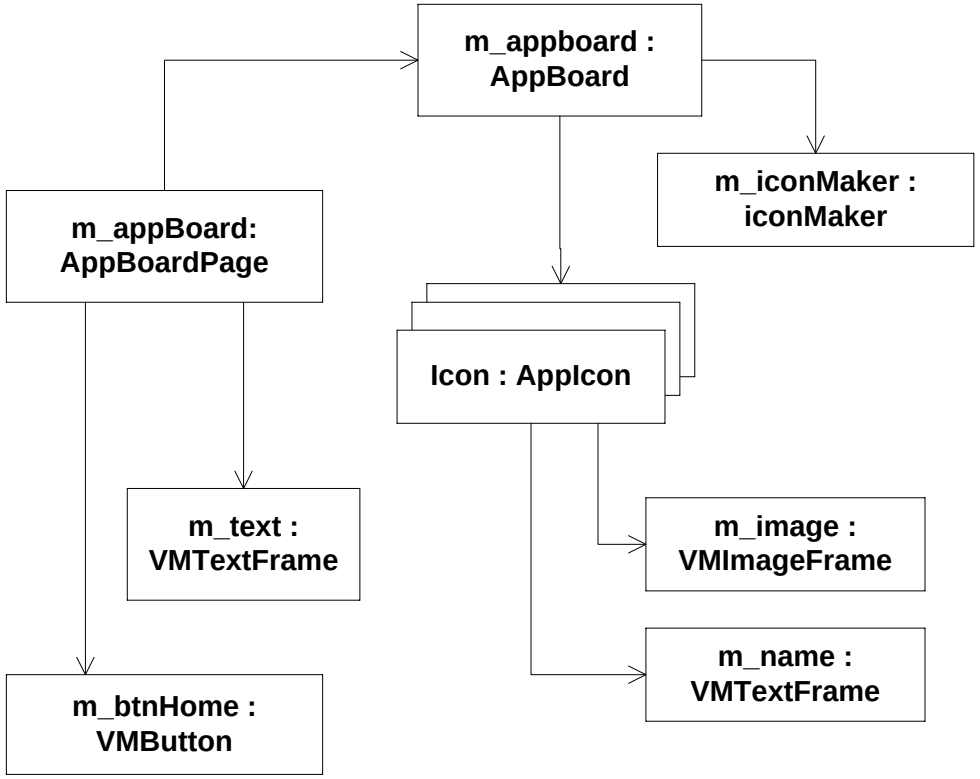
- Hello World Explanation



Launcher Provider – Hello World

- Hello World Explanation





MEDIATEK

www.mediatek.com