# UIKit lib basic Integration

To support backward compatibility of different UI components, UIKitLIb uses appcompat lib. So the base activity always extends from AppCompatActivity.

Documentation: Open index.html file from JavaDoc.zip and search for MainActivity. MainActivity provides all the components Javadoc link.

## Steps to integrate:

Before using any feature, the lib must be included in the project.

The recommended way is to have artifactory dependency pointing to uikitlib.

Online build:

compile 'com.philips.cdp:uikitLib:3.0.1'   (works in Philips network)

Using aar file:

Add below dependency in your app build.gradle and place aar files in libs folder

compile **'com.android.support:support-annotations:22.2.0'**compile **'com.android.support:appcompat-v7:23.1.1'**compile **'uk.co.chrisjenx:calligraphy:2.1.0'**compile **'com.android.support:design:23.1.1'**compile **'com.shamanland:fonticon:0.1.8'**compile **'com.android.support:cardview-v7:+'**compile **'com.android.support:recyclerview-v7:+'**

Add below line inside defultConfig in app build.gradle file

defaultConfig {

generatedDensities = []  
}

Artifactory link . Include maven dependency in project build.gradle(not app)

allprojects {  
 repositories {  
 maven {  
 url 'http://maartens-mini.ddns.htc.nl.philips.com:8081/artifactory/libs-release-local-android'  
 }  
 }  
}

Make sure you use http**s** proxy when downloading library or else it would not download.

### Choose theme: Lib support two variants of themes (Solid and gradient version).

Various themes can be accessed with Theme.Philips.\* and Theme.Philips.\*.Gradient

Example: Blue theme Solid-> **Theme.Philips.DarkBlue**

Gradient-> **Theme.Philips.DarkBlue.Gradient**

### Extend from UIKitActivity-> UIKitActivity provides some common customizations to achieve some goals like having common fonts across all activities.

Example: **public class** MyActivity **extends** UiKitActivity {}

### Using custom fonts:

UIKit uses open source lib [Calligraphy](https://github.com/chrisjenx/Calligraphy) for font customization.

Define fontpath=”fonts/myfont.ttf” in xml for text view to use custom font.

### VectorDrawable Support: Custom <vector> tag is recognized only on or after Lollipop. Vector Drawable is taken from AOSP version 5.0.0\_r2

But still it can be used via programmatically.

Example: -> Drawable d =

com.philips.cdp.uikit.drawable.VectorDrawable.create(context, R.drawable.xyz);

imageView.setImageDrawable(d);

1. Set Theme:

Apply a theme to an Activity or application

To set a theme for all the activities of your application, open the AndroidManifest.xml file and edit the <application> tag to include the UIKit theme attribute with the style name. For example:

<application android:theme=" @style/Theme.Philips.BrightAqua">

If you want a theme applied to just one Activity in your application, then add the android:theme attribute to the <activity> tag instead.

<activity android:theme="@style/Theme.Philips.BrightAqua ">

Theme can set through code

Example: for BrightBlue theme

setTheme(R.style.Theme\_Philips\_BrightBlue);

UiKit supporting 18 themes

Theme.Philips.DarkBlue

Theme.Philips.DarkPurple

Theme.Philips.DarkGreen

Theme.Philips.DarkOrange

Theme.Philips.DarkAqua

Theme.Philips.DarkPink

Theme.Philips.BrightOrange

Theme.Philips.BrightAqua

Theme.Philips.BrightGreen

Theme.Philips.BrightPink

Theme.Philips.BrightBlue

Theme.Philips.BrightPurple

Theme.Philips.LightBlue

Theme.Philips.LightOrange

Theme.Philips.LightAqua

Theme.Philips.LightGreen

Theme.Philips.LightPink

Theme.Philips.LightPurple

1. Model alert

To support model alert include below script under defaultConfig on application build.gradle file

renderscriptTargetApi 23

renderscriptSupportModeEnabled true