UNIVERSITÀ
DEGLI STUDI
DE L'AQUILA

disim

Mobile Programming Laboratory

ANDROID
Resources and Qualifiers

Teachers

Ing. Tarquini Francesco, Ph.D

Ph.D in Computer Science Engineering

francesco.tarquini@univaq.it

Ing. D'Errico Leonardo

Ph.D Student in Computer Science Engineering

leonardo.derrico@graduate.univaq.it

Teaching Materials

Available on MOODLE platform

http://www.didattica.univaq.it

Google Drive Repository

https://drive.google.com/drive/folders/1ISqZfn0i9Ub3eWNXbvW00rd0hD9ya8OL?usp=sharing

Topics

- Resources
  - Colors
  - Strings
  - Drawables
  - Other
  - Some Rules
  - Referencing
- Qualifiers
  - Internationalization and localization (i18N)
  - Screen Density
  - Other qualifiers

Resources

In every Android project exists a **res** folder including all project resources.

Images, Layouts, Strings, Colors, Styles, Fonts, Audio, Video and so on.

Android Studio organizes the resources in a set of folder in according with the resources types:

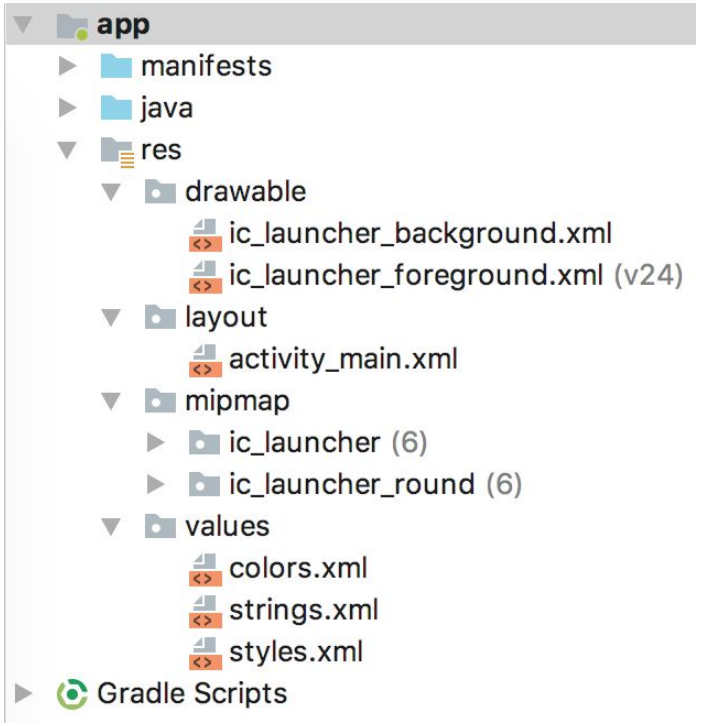
Drawable: images or everything is drawable as Shape

Layout: graphics screen description

Mipmap: the application icon

Values: application Strings, dimension values, colors and styles

Every resources can be qualified in according to the size, language, android version, location, and so on, of the device where the application runs.

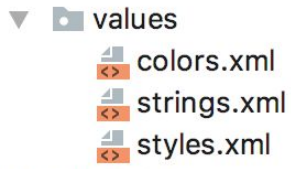


Resources - Colors

In Android the colors are defined in hexadecimal format with pattern:

#ARGB or **#AARRGGBB** (also without Alpha)

The developer can declare the color as value of the view attributes or if he create a palette of colors he can declare it inside a **XML file** in **values** folder.



The IDE create automatically in the HelloWorld project a file named **colors.xml** in values folder.

The colors are defined as a map where the ARGB value is mapped in a String name.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <color name="colorPrimary">#3F51B5</color>
  <color name="colorPrimaryDark">#303F9F</color>
  <color name="colorAccent">#FF4081</color>
</resources>
```

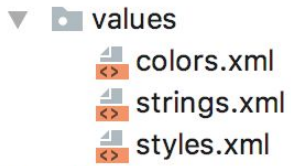
Resources - Strings

Every text string of the our project can be directly added as a value of the **text attribute** of the views: TextView, Button, … .

This is allowed but it is **against** of the **I18N** rules (Internationalization).

The right way to use a String is to create a **strings.xml** file in values folder and define avery string in this file *[you could have more than one file]*.

The application name is also a resource of String type.




```
<resources>
  <string name="app_name">HelloWord</string>
</resources>
```

Resources - Drawables

Inside the drawable folder the developer can add the picture of the application. The only allowed pictures are: png (*.9.png too), jpg, or gif.

In this folder it is also possible to create some XML file to define:

Layer List: a Drawable that manages an array of other Drawables. These are drawn in array order, so the element with the largest index is be drawn on top. Creates a LayerDrawable.

State List: an XML file that references different bitmap graphics for different states (for example, to use a different image when a button is pressed). Creates a StateListDrawable.

Level List: an XML file that defines a drawable that manages a number of alternate Drawables, each assigned a maximum numerical value. Creates a LevelListDrawable.

Transition Drawable: an XML file that defines a drawable that can cross-fade between two drawable resources. Creates a TransitionDrawable.

Resources - Drawables

Inset Drawable: an XML file that defines a drawable that insets another drawable by a specified distance. This is useful when a View needs a background drawable that is smaller than the View's actual bounds.

Clip Drawable: an XML file that defines a drawable that clips another Drawable based on this Drawable's current level value. Creates a ClipDrawable.

Scale Drawable: an XML file that defines a drawable that changes the size of another Drawable based on its current level value. Creates a ScaleDrawable

Shape Drawable: an XML file that defines a geometric shape, including colors and gradients. Creates a GradientDrawable.


Resources - Others

In the previous slides we illustrated some of the main resources, but more other resources exist.

The **audio** resources as **wav**, **mp3**, **aac**, **flac**, and so on, can be added in the resource folder named **xml** that not exists to default but the developer can create it.

The **preference** resource help the developer to create easily a settings screen. These resources are added always in **xml** folder.

Since the last year Google add the **font** resource, too. In this case the fonts are added in a **font** folder and configured in **xml** folder.

```
<?xml version="1.0" encoding="utf-8"?>
<font-family xmlns:android="http://schemas.android.com/apk/res/android">
  <font
     android:fontStyle="normal"
     android:fontWeight="400"
     android:font="@font/fontname" />
</font-family>
```

Resources - Some Rules

The only characters allowed for the name of the resources must follow the pattern **a-z0-9** and as special character is just allowed the underscore “**_**”

The resources name **must** start with a letter

Resources - Referencing

If the developer want use a resource, he can reference this last using the follow way:

In **XML** file the **@ character** give us the access to the resources:

@<resource type>/<resource name>

In **Java** file the **R class** give us the access to the resources:

R.<resource type>.<resource name>

As resource type we can have:
color, dimen, drawable, id, layout, menu, mipmap, menu, string, style, xml

Resources - Referencing

@<resource type>/<resource name> **R.<resource type>.<resource name>**

As resource name you have:

color, dimen, string, style: the key of the map where it is defined

layout, menu, drawable, preference: the name of the file without its extension

id: the name of the id defined after @+id/<id name>

Qualifiers

In the world more different devices exist and its variety mainly consists in:

languages, **android version**, **density, size or format of the display, orientation**

The developer cannot design an application for every specific device, thus Android introduces the **qualifiers** to specialize all resources.

In detail, the developer must not qualify every single resource but must qualify the folder, so every added resource in this last folder is qualified.

Every folder can have **one or more** qualifiers.

Be Careful! It is recommended add for every qualified resource one of this inside a folder without qualifiers.

Qualifiers - Internationalization and localization

An example of qualifier is that to internationalize and/or localize the application.

If the developer managed the strings of the application in the strings.xml file, he can change the language of his application using the qualifiers:

Italian language:
a translated copy of **strings.xml** must be added in a **values-it** folder that the developer must create.

Italian location:
Italian language is spoken in Italy but also in Republic of San Marino and in Switzerland, if the developer want just locate the italian strings in Switzerland, he must be have a **values-it-rCH** folder (**values-it-rIT** for Italy)

Be Careful! It is recommended include in the default folder the strings in english.

Qualifiers - Screen Density

Android include some screen density qualifiers.

The density qualifiers are differentiated in:

Low (**ldpi** ~120dpi) [deprecated]
Medium (**mdpi** ~160dpi) is the default qualifier. It is the baseline
High (**hdpi** ~240dpi): the ratio is 1.5 times the baseline
Extra-high (**xhdpi** ~320dpi): the ration is x2 the baseline
Extra-extra-high (**xxhdpi** ~480dpi): the ration is x3 the baseline
Extra-extra-extra-high (**xxxhdpi** ~640dpi): the ration is x4 the baseline

In bold the name of the qualifiers.

Example: **drawable-xxhdpi** or **layout-xhdpi**

Qualifiers - Other qualifiers

Android give us much more qualifiers and a tool to add them.

It is recommended that the developer uses this tool to add multiple qualifiers in the right order.

Right click on res folder and select
New > Android Resource File
to show the tool.