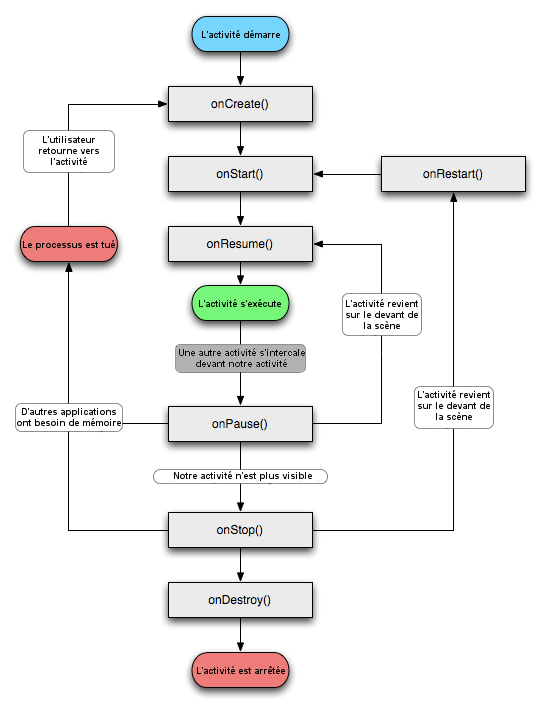
Projet libre Android

# Contexte

On codera en Android version 8 pour être accessible à 95% ( ?) des utilisateurs sur le marché, mais on compilera dans la version 17 pour l’optimiser sur les téléphones à jour.

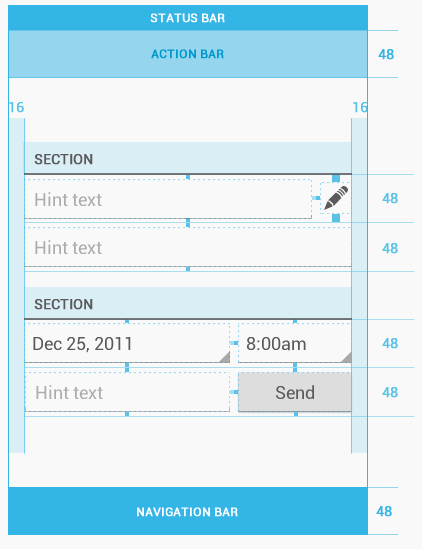
Nous avons fait attention à laisser la priorité aux appels, etc :



Design de l’interface

* Utiliser des icônes au lieu des boutons (login, …)
* Laisser les gens personnaliser l’interface, icône user, et s’en rappeler d’une connexion à l’autre !
* Utiliser auto-complétion, proposer derniers choix dans les barres de recherche
* Retour arrière en cas d’erreur ! Faire une architecture claire de la progression dans le programme.
* Le plus simple (ex étoile : barres de slider et modification instantanée)
* Faire du feedback (commentaire bien chargé ! ou autre)
* Le mouvement de « swipe » est intuitif : entre différents commentaires/notations d’UE (plus les petits points en haut pour se retrouver et titre global) ? + raccourcis
* Messages d’erreur (champs non remplis)
* Raccourcir toutes les tâches (pas de gros messages pleins d’erreurs), et remarque quand c’est bien fait (check box vert, glow, …)
* Mettre en priorité ce qui est important
* Messages d’explication (qu’on peut dismiss si on connait)
* Ajouter des couleurs sur les touches pressées/non
* Ajouter des couleurs : commentaires non lus ?
* Actions et contenu dès le premier écran, éviter les redirections

Au moins 48 dp ! Et 8 dp entre chaque élément :



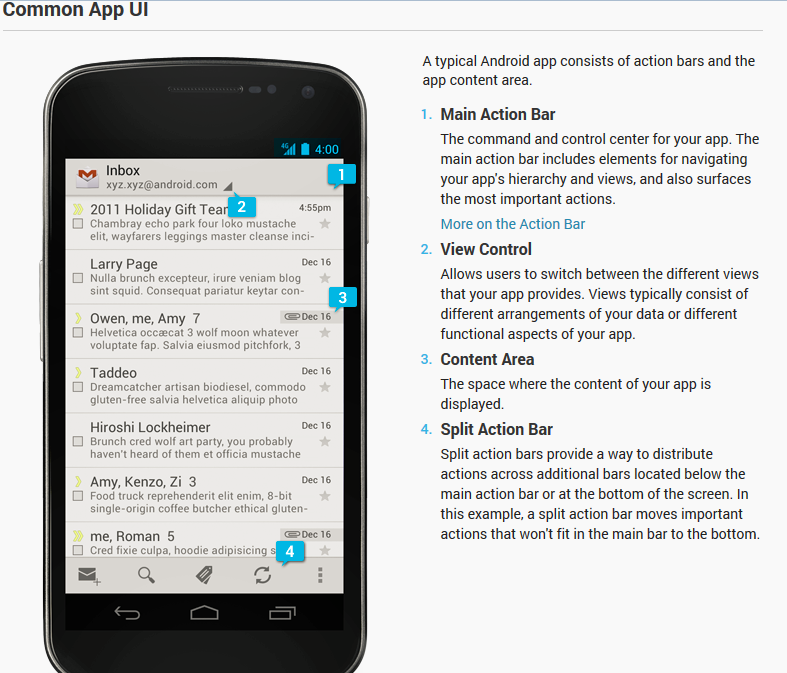
Caractères : roboto – light/regular si API >= 14

Couleur : #FFBB33 ou #FF8800 (<http://developer.android.com/design/style/color.html>)

Icone : launcher 512x512px et 48dp, action 24x24/32x32dp (cf. dossier), small 12x12/16x16dp, notification 22x22/24x24dp

MAX 30 CARACTERES PAR PHRASE

Bug : MyApp isn't responding. Do you want to close it?



Notifications ?

Thème : Holo Light

Plusieurs écrans (grands écrans => plus de contenu) ?

En bas : …/MaJ/ Recherche/ login/Parametres

Action bar : Colors: **#333333,** Enabled: **60%** opacity, Disabled: **30%** opacity (ex. Ajouter commentaire, ..)

Coder interface (boutons)

# Dans le XML

<?xml version=*"1.0"* encoding=*"utf-8"*?>

<LinearLayout xmlns:android=*"http://schemas.android.com/apk/res/android"*

xmlns:tools=*"http://schemas.android.com/tools"*

android:layout\_width=*"match\_parent"*

android:layout\_height=*"match\_parent"*

android:paddingBottom=*"@dimen/activity\_vertical\_margin"*

android:paddingLeft=*"@dimen/activity\_horizontal\_margin"*

android:paddingRight=*"@dimen/activity\_horizontal\_margin"*

android:paddingTop=*"@dimen/activity\_vertical\_margin"*

android:orientation=*"vertical"*

tools:context=*".MainActivity"* >

<TextView

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"wrap\_content"*

android:text=*"Taille : "*

android:textStyle=*"bold"*

android:textColor=*"#FF0000"*

android:gravity=*"center"*

/>

<EditText

android:id=*"@+id/taille"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"wrap\_content"*

android:hint=*"Taille"*

android:inputType=*"numberDecimal"*

/>

<CheckBox

android:id=*"@+id/mega"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:text=*"Mega fonction !"*

/>

<Button

android:id=*"@+id/calcul"*

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:text=*"Calculer l'IMC"*

/>

<TextView

android:layout\_width=*"wrap\_content"*

android:layout\_height=*"wrap\_content"*

android:text=*"Résultat:"*

/>

<TextView

android:id=*"@+id/result"*

android:layout\_width=*"fill\_parent"*

android:layout\_height=*"fill\_parent"*

android:text=*"Cliquez sur le bouton pour obtenir un résultat."*

/>

</LinearLayout>

# Dans l’activité Java (src)

package com.shapter;

import android.os.Bundle;

import android.app.Activity;

public class MainActivity extends Activity {

private final String defaut = "Cliquez sur le bouton pour obtenir un résultat.";

private final String megaString = "Vous faites un poids parfait !”;

Button envoyer = null;

EditText taille = null;

CheckBox mega = null;

TextView result = null;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

// On récupère toutes les vues dont on a besoin

envoyer = (Button)findViewById(R.id.calcul);

result = (TextView)findViewById(R.id.result);

taille = (EditText)findViewById(R.id.taille);

mega = (CheckBox)findViewById(R.id.mega);

// On attribue un listener adapté aux vues qui en ont besoin

envoyer.setOnClickListener(envoyerListener);

taille.addTextChangedListener(textWatcher);

mega.setOnClickListener(checkedListener);

}

private TextWatcher textWatcher = new TextWatcher() {

@Override

public void onTextChanged(CharSequence s, int start, int before, int count) {

result.setText(defaut);

}

@Override

public void beforeTextChanged(CharSequence s, int start, int count,

int after) {

}

@Override

public void afterTextChanged(Editable s) {

// TODO Auto-generated method stub

}

};

// Listener du bouton de la megafonction.

private OnClickListener checkedListener = new OnClickListener() {

@Override

public void onClick(View v) {

// On remet le texte par défaut si c'était le texte de la megafonction qui était écrit

if(!((CheckBox)v).isChecked() && result.getText().equals(megaString))

result.setText(defaut);

}

};

private OnClickListener envoyerListener = new OnClickListener() {

@Override

public void onClick(View v) {

result.setText(megaString);

}

};

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.main, menu);

return true;

}

}

<http://developer.android.com/training/basics/firstapp/building-ui.html>

(pour le code de l’interface)

<http://developer.android.com/design/get-started/creative-vision.html>

(Pour le design)