Menus & Action Bar

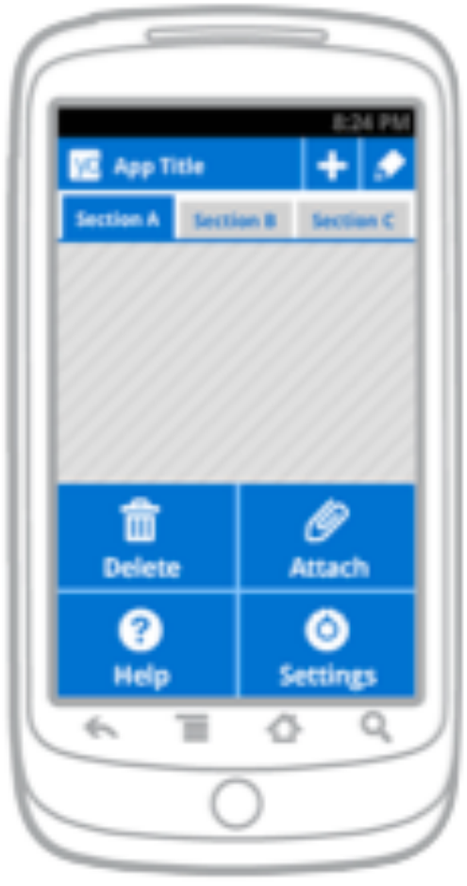
Jean-Marc Lecarpentier
Université de Caen

Au programme

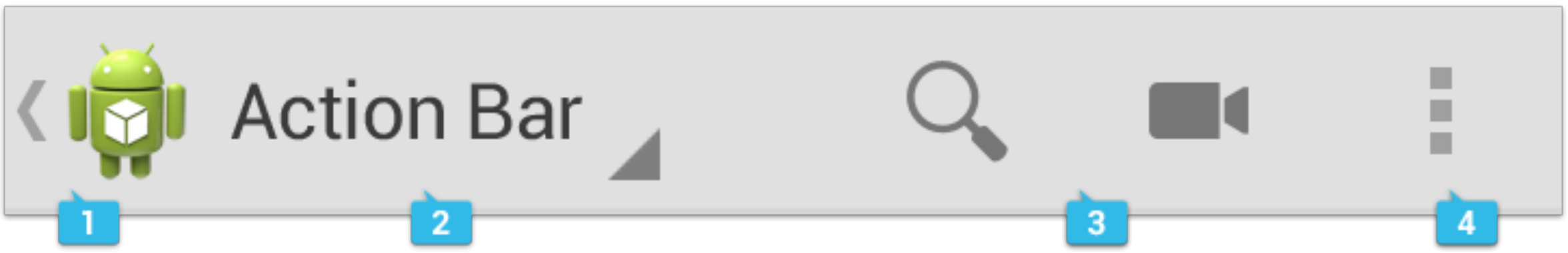
- Navigation dans une App
- Barre d’actions Android
- Structure & gestion de la barre
- Menus Contextuels

Action Bar

- Centralise les menus et boutons
- Permet de gagner de la place
- Homogénéité de l’expérience utilisateur sur toutes les applications
- Gestion du placement des menus en fonction de la taille écran



Structure



1. Icone : avec ou sans navigation <u>dans l'application</u>

2. Titre de l'application et éventuel menu

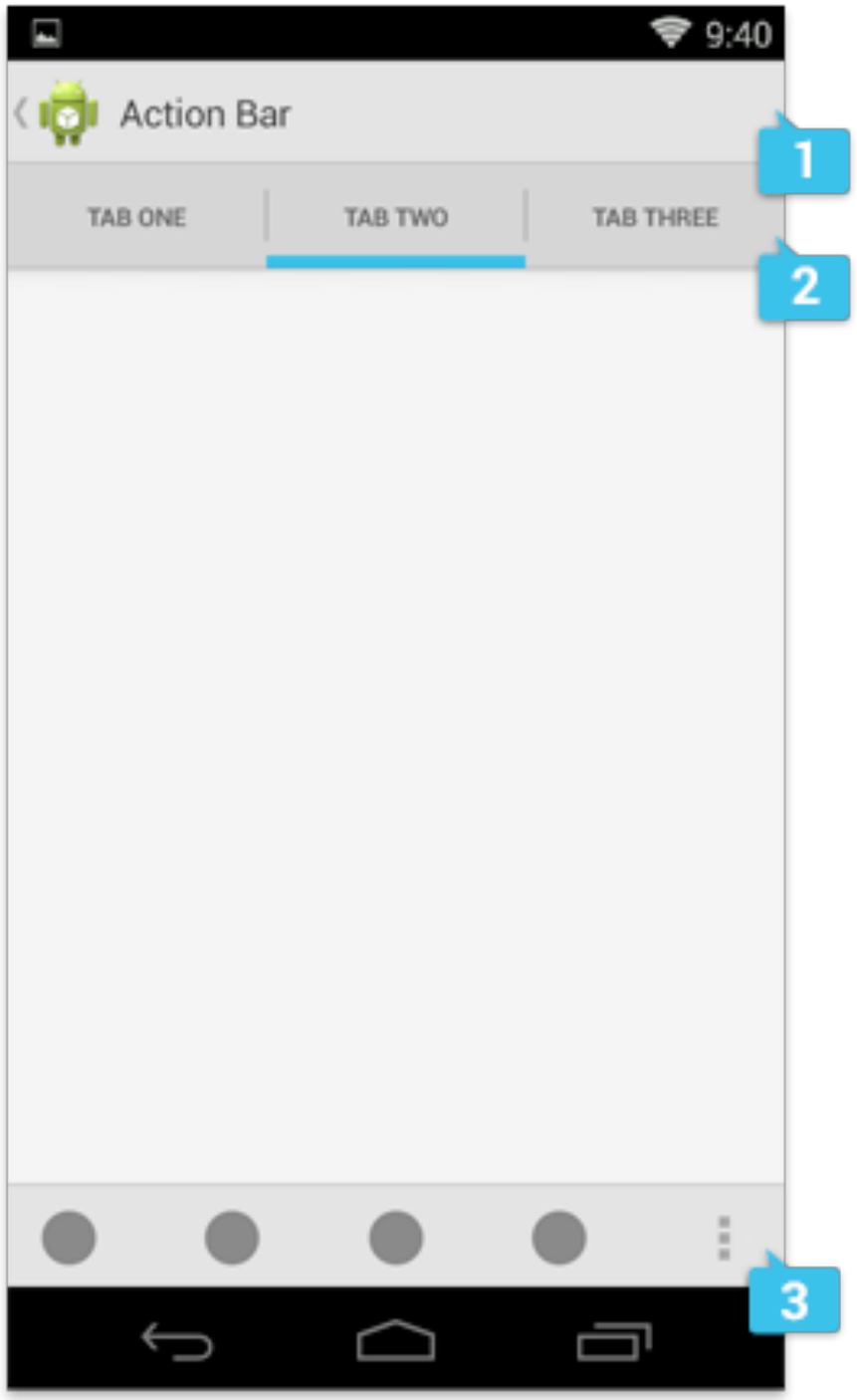
3. Boutons pour lancer les actions principales

4. *Action overflow* : Menu contenant les actions moins souvent utilisées

Structure en étages

- Possibilité de mettre les actions sur plusieurs niveaux

    1. Main action bar
    2. Top bar
    3. Bottom bar



Création de l'ActionBar

- Gérée par l'Activity en cours d'exécution
- Activity doit dériver de ActionBarActivity
- Thème doit inclure la barre
- API ≥ 11: barre incluse par tout thème dérivant de Theme.Holo
- Barre présente par défaut

Manipulation de la barre

- Dans l'activité :
  ActionBar actionBar = getActionBar();

- Possibilité de montrer et cacher la barre :
  actionBar.show();
  actionBar.hide();

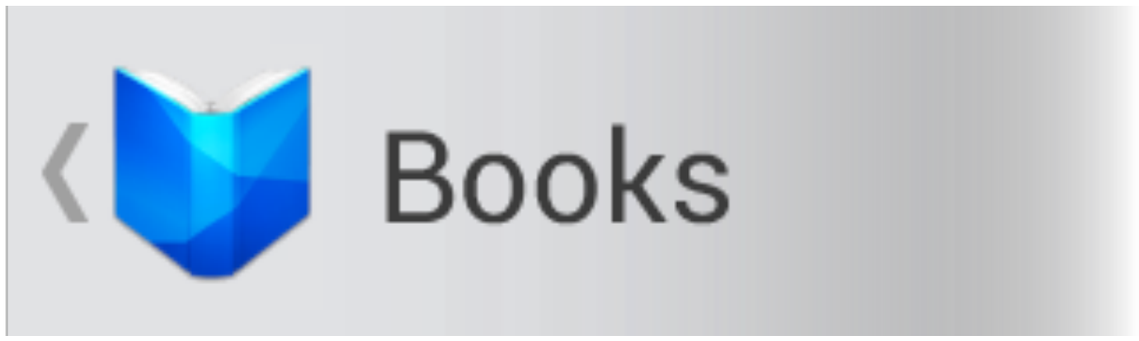
Icone de la barre

- Icone par défaut = icone de l'App

- Utiliser un logo à la place : attribut logo de l'élément <application> ou <activity> dans le Manifest

- Activer l'icone comme bouton *Up* :
  ActionBar actionBar = getActionBar();
  actionBar.setDisplayHomeAsUpEnabled(true);

Back vs. Up



- Back = gestion par le système
- Navigation par ordre chronologique inverse des écrans
- Relation temporelle entre écrans
- Indépendant de la hiérarchie de l’application

- Navigation dans la hiérarchie de l’application
- Écran d’accueil sans bouton Up

http://developer.android.com/design/patterns/navigation.html

Spécifier le “parent”

- Navigation *Up* : chaque Activity peut préciser son activité “parente”

- Dans le Manifest (cas général) :

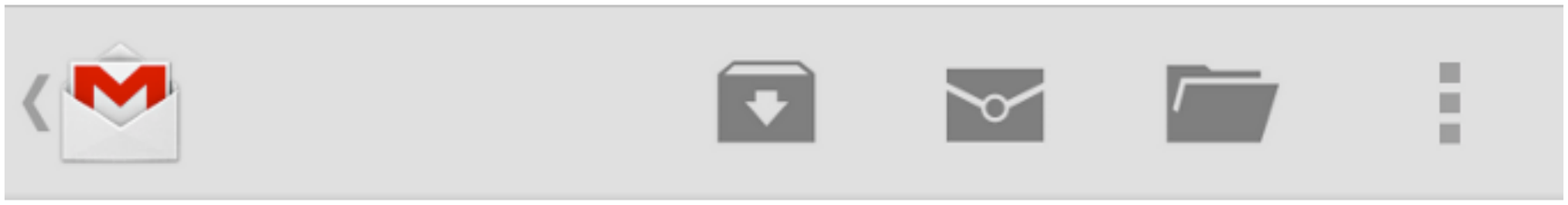
```
<activity
    android:name="com.example.myfirstapp.DisplayMessageActivity"
    android:label="@string/title_activity_display_message"
    android:parentActivityName="com.example.myfirstapp.MainActivity" >
```

- Avec du code (si le parent dépend du contexte) :

```
getParentActivityIntent();
```

http://developer.android.com/training/implementing-navigation/ancestral.html

Boutons d’actions



- Mettre les actions les plus importantes en premier
- Système place automatiquement en fonction de la place disponible
- Liste définie dans un layout XML avec <menu>
- Boutons mis en place par onCreateOptionsMenu

Menu XML de la barre

- Fichier res/menu/accueil_action_bar.xml

```
<menu xmlns:android="http://schemas.android.com/apk/res/android" >
    <item android:id="@+id/action_search"
          android:icon="@drawable/ic_action_search"
          android:title="@string/action_search"/>
    <item android:id="@+id/action_compose"
          android:icon="@drawable/ic_action_compose"
          android:title="@string/action_compose" />
</menu>
```

- Spécifier si un bouton doit apparaître ou non, Attribut showAsAction :

```
android:showAsAction=["ifRoom" | "never" | "withText" | "always" | “collapseActionView"]
```

- Mettre android:showAsAction=“ifRoom" pour le faire apparaitre
- Toujours mettre icon ET titre
- Éviter “always”

Mise en place de la barre

```
public boolean onCreateOptionsMenu(Menu menu) {
    // Mettre l'action bar
    MenuInflater inflater = getMenuInflater();
    // XML du fichier accueil_action_bar
    inflater.inflate(R.menu.accueil_action_bar, menu);
    return super.onCreateOptionsMenu(menu);
}
```

Capter un clic dans la barre

- Callback onOptionsItemSelected (MenuItem item)

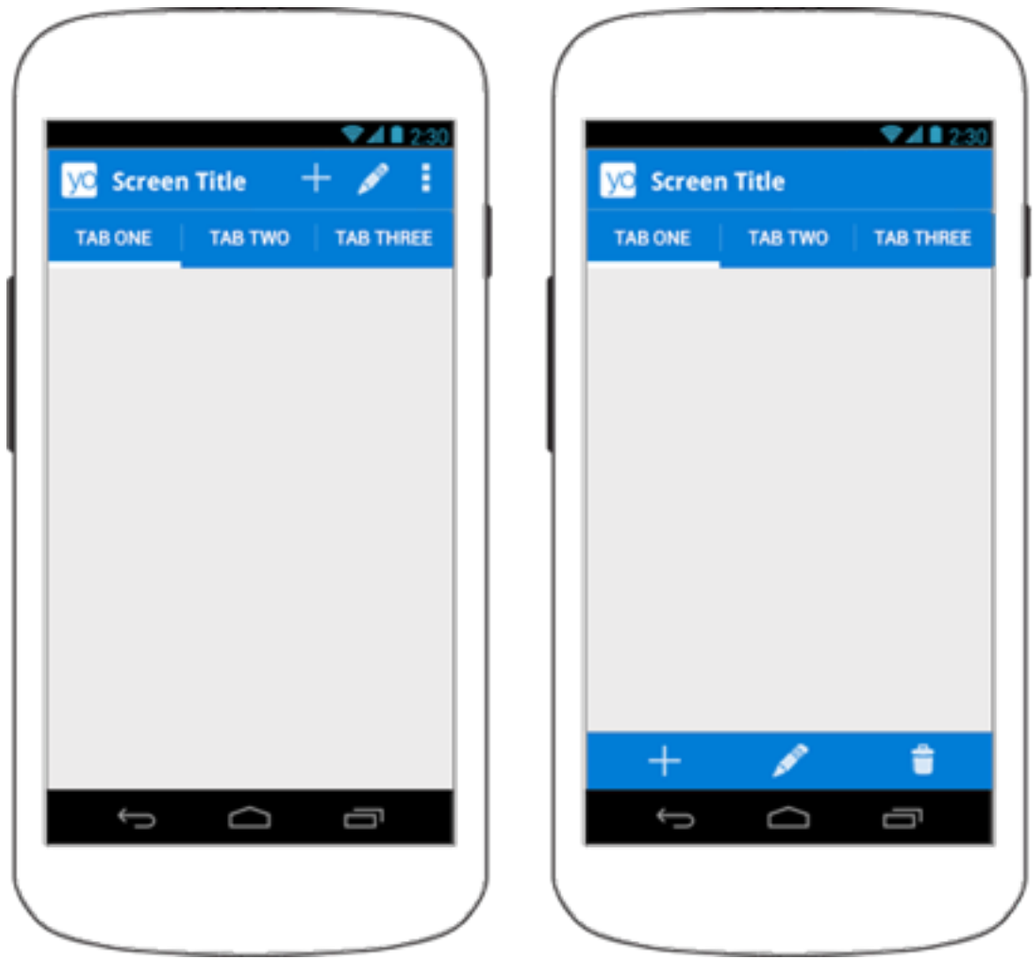
```
public boolean onOptionsItemSelected(MenuItem item) {
    // Capter le clic, que faire ?
    switch (item.getItemId()) {
        case R.id.action_search:
            openSearch();
            return true;
        case R.id.action_compose:
            composeMessage();
            return true;

        // Passer à la classe parente pour le cas des fragments
        default:
            return super.onOptionsItemSelected(item);
    }
}
```

ActionBar à “étages”

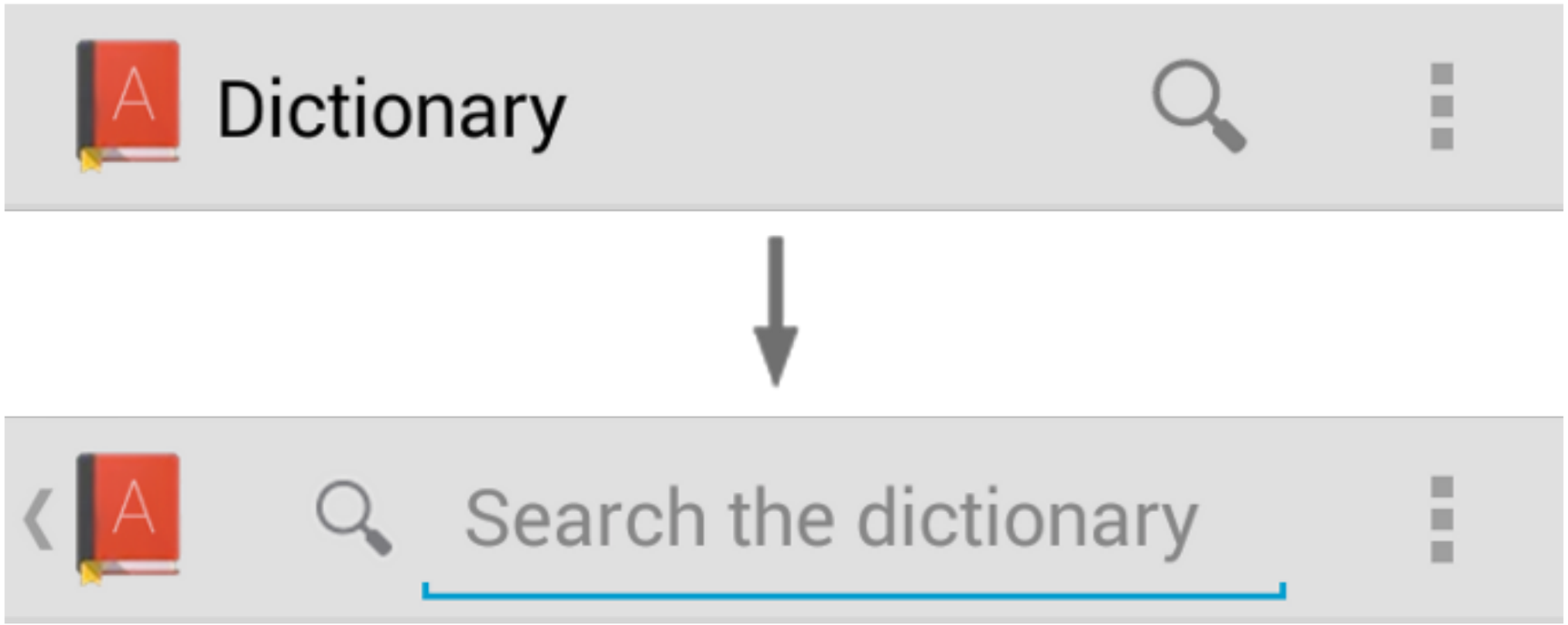
- Gestion automatique en fonction de la taille et de l’orientation de l’écran

- Déclarer dans le Manifest :

```
<activity
  uiOptions="splitActionBarWhenNarrow"
  ...
>
```



Action Views



- Permettre d’ajouter des éléments d’interaction dans la barre

- Exemple typique : Recherche

Action Views

- Fichier XML

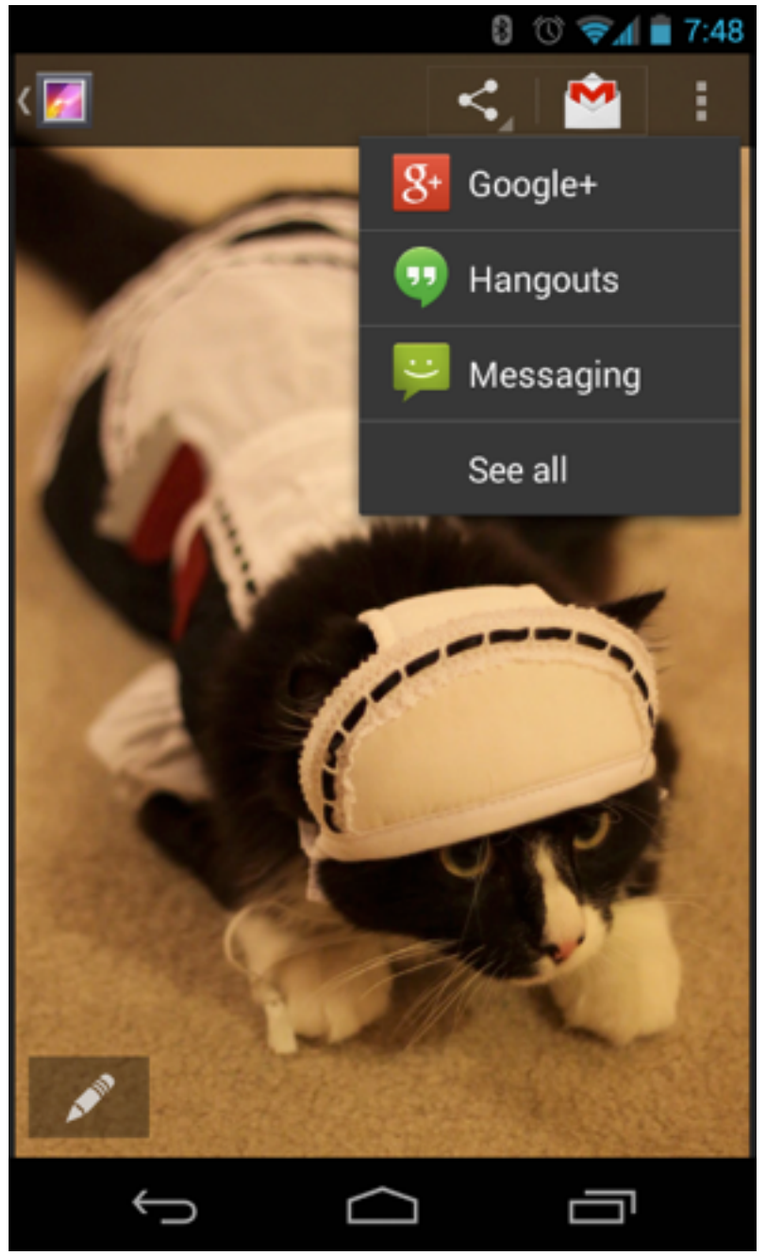
```
<item android:id="@+id/action_search"
      android:title="@string/action_search"
      android:icon="@drawable/ic_action_search"
      android:showAsAction="ifRoom|collapseActionView"
      android:actionViewClass="android.widget.SearchView" />
```

- Gérer le comportement de la recherche

```
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.main_activity_actions, menu);
    SearchView searchView = menu.findItem(R.id.action_search).getActionView();
    // Configurer la recherche et ses capteurs d'évènements
    ...
    return super.onCreateOptionsMenu(menu);
}
```

Ajouter des sous-menus

- Un bouton de la barre peut ouvrir un sous-menu
- Objet ActionProvider
- Créer ses ActionProvider
- Utiliser prédéfinis, par exemple “Share”



Action Provider

- Fichier XML

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:yourapp="http://schemas.android.com/apk/res-auto" >
    <item android:id="@+id/action_share"
          android:title="@string/share"
          android:showAsAction="ifRoom"
          android:actionProviderClass="“android.widget.ShareActionProvider"
          />
    ...
</menu>
```

- Gérer le comportement

```
private ShareActionProvider mShareActionProvider;
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.main_activity_actions, menu);

    // Metre un Intent part défaut pour initialiser le bouton ActionProvider
    mShareActionProvider = menu.findItem(R.id.action_share).getActionProvider();
    mShareActionProvider.setShareIntent(getDefaultIntent());

    return super.onCreateOptionsMenu(menu);
}

/** Définir une action par défaut, mais celle-ci devra éventuellement être redéfinie losrque l'on sait exactement l'objet manipulé
  * mShareActionProvider.setShareIntent()
  */
private Intent getDefaultIntent() {
    Intent intent = new Intent(Intent.ACTION_SEND);
    intent.setType("image/*");
    return intent;
}
```

En bref

- Utiliser la barre d'action pour gérer la navigation et les actions dans son application

- Gestion de l'affichage en fonction de la taille d'écran et de son orientation

- Barre d'action contextuelle (CAB)

- Styler la barre et Créer des icones pour la barre

- Pour en faire plus :
  Utilisation d'onglets
  Tuto Onglets
  Menus DropDown