ActionBar et ToolBar

Jeremy MATHIAS (FA)
TP le 19/10/2017

Résumé

Depuis la version 3.0 d'android, la barre d'action (ActionBar) a fait son apparition afin de proposer un visuel cohérent entre les applications android car elle est présente dans les thèmes par défaut des applications. L'ActionBar permet à l'utilisateur d'accéder simplement aux informations utiles de l'application (titre, menu d'action, etc). La version 5.0 d'android a, quant à elle, apporté un nouveau composant: la toolBar. Il s'agit d'une généralisation de l'ActionBar.

Dans ce TP, nous verrons comment créer une ActionBar et une ToolBar, quelles sont les différences entre ces deux éléments et quel choix faire pour les applications actuelles.

Pré-requis

- Maîtriser les layouts
- Savoir programmer une application android contenant plusieurs activités

Code source

Code source initial disponible ici : https://github.com/munggs/ActionBar-ToolBar-initial

Code source final disponible ici : https://github.com/munggs/ActionBar-ToolBar-final

Explications du TP

ActionBar

Une **ActionBar** peut se composer de 4 parties :

- **L'icone de l'application** : Etablie l'identité visuelle de l'application et permet aussi de naviguer dans l'application (remonter à la vue mère par exemple).
- **Dropdown Menu** : Permet d'afficher le titre de la vue actuelle ainsi que naviguer dans l'application (exemple de l'application Gmail).
- **Actions principales** : Définie les actions principales de votre application.
- **Autres actions** : Permet d'accéder aux fonctionnalités moins importantes de l'application.

Etape 1 : Créer une activité sans barre d'action

Lancer l'application pour constater qu'il n'y a aucune barre d'action sur l'activité. Pour faire cela, il suffit d'utiliser un thème dont le parent ne possède pas d'actionBar. (*Theme.AppCompat.Light.NoActionBar* dans notre cas.)

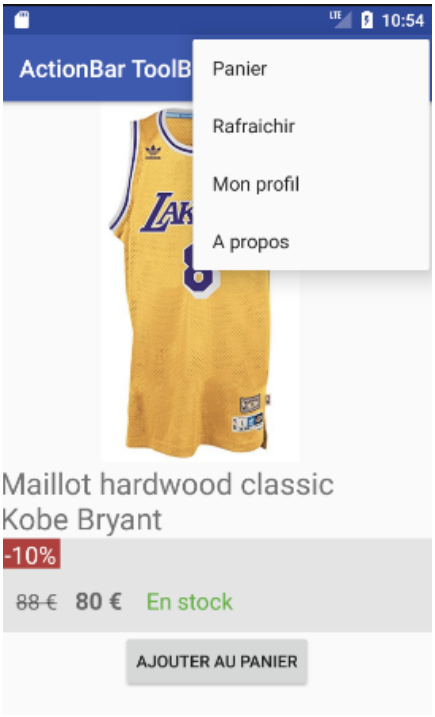
Etape 2 : Ajouter une actionBar à l'activité

Pour ajouter une actionBar à l'activité, il faut donc d'utiliser un thème ayant une ActionBar:
Dans styles.xml : ajoutez une action bar en retirant le commentaire sans oublier d'ajouter un commentaire à la ligne du dessus.

Etape 3 : Ajouter des actions secondaires à l'actionBar dans un menu

Afin d'ajouter un menu et des actions à notre ActionBar, il faut d'abord créer un fichier xml dans res/menu et y déclarer les items représentant les actions. Pour cela,

décommentez les 4 items du fichier *res/menu/app_menu.xml*. Il ne vous reste ensuite plus qu'à « injecter » ce composant au menu de l'actionBar : Décommenter la méthode *onCreateOptionsMenu* de *MainActivity.java*.



Etape 4 : Ajouter des actions principales à l'actionBar

Comme expliqué précédemment, chaque actions (principales ou secondaires) de l'actionBar est décrite dans une balise item du menu. Pour définir s'il s'agit d'actions principales ou secondaires, on utilise l'attribut *showAsAction.*

Il peut prendre les valeurs suivantes :

- ifRoom -> élément ajouté aux actions principales de l'action bar si il y a de la place, sinon, il sera présent dans les actions supplémentaires (trois petits points).
- always -> élément toujours présent dans les actions principales (peut causer des superpositions d'éléments, à utiliser avec précautions).
- never -> élément jamais présent dans les actions principales et toujours dans les actions secondaires.
- withText -> texte de l'élément présent s'il y a de la place.

Pour tester :

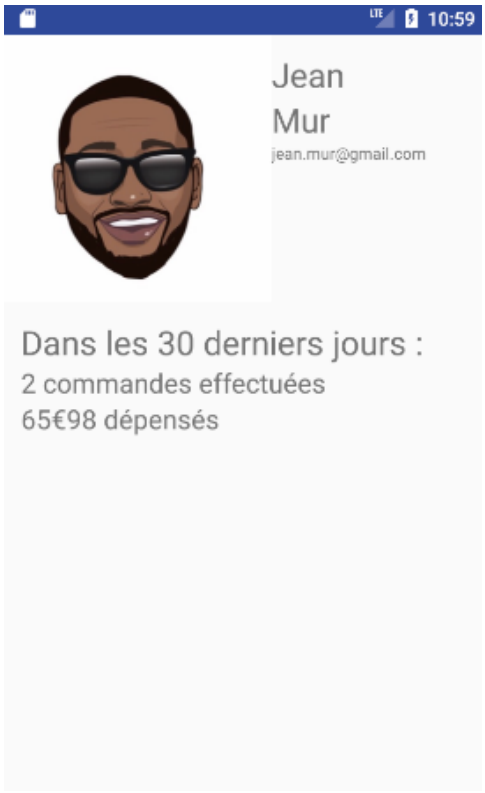
Remplacez la valeur (never) de l'attribut showAsAction par :

- always pour l'item de panier.
- ifRoom pour l'item de rafraichissement.
- ifRoomlwithText pour l'item de profil.

Etape 5 : Prendre en compte le click sur ces actions

Les différentes actions sont maintenant présentes mais elles ne sont pas encore fonctionnelles. La prise en compte du click se fait dans la méthode *onOptionsItemSelected* où on peut définir le comportement de chacun des items grâce à leur id.

Decommentez la méthode onOptionsItemSelected de MainActivity.java.



Etape 6 : Afficher la valeur d'un compteur dans l'actionBar

Nous souhaitons maintenant voir le nombre d'articles ajouter au panier dans la barre d'action. Pour cela, on créé une superposition d'item grâce à la balise *layer-list* du

fichier *ic_menu_cart.xml* . Quand on clique sur le bouton d'ajout au panier, on incrémente la valeur du compteur et on invalide les options du menu pour forcer un rafraichissement de la barre d'action. Et, lors de la création du menu, on récupère la valeur du compteur et on dessine le « badge » représentant le compteur sur l'item du panier.

Décommentez le if dans la méthode *onCreateOptionsMenu* de la classe *MainActivity*

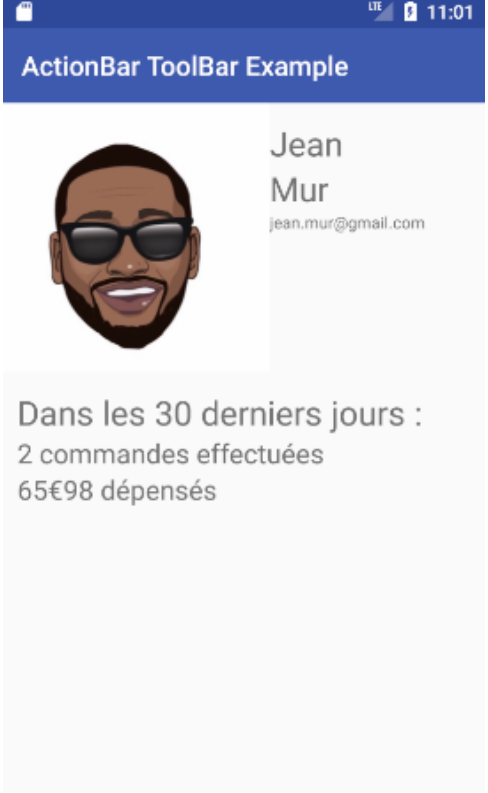
Etape 7 : Créer une ToolBar en tant qu'actionBar

On peut maintenant accéder à l'activité du profil qui ne contient pas d'actionBar. Regardez dans styles.xml que le thème utilisé est en NoActionBar

Nous allons y ajouter une ToolBar. La toolbar est une généralisation de l'actionBar, et, contrairement à celle-ci, est définit dans le layout de l'activité. La toolBar est donc accessible comme n'importe quel élément d'un layout.

Décommentez la déclaration de la toolbar dans activity_profile.xml.

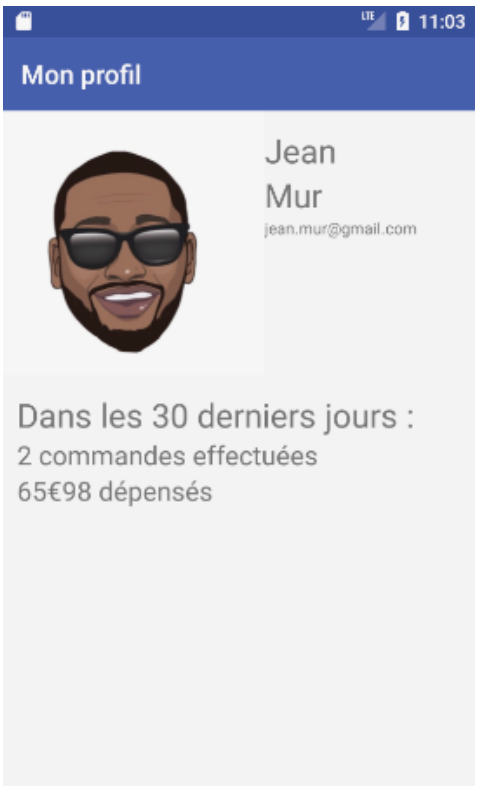
Décommentez la déclaration de la toolbar en tant qu' « actionbar » dans ProfileActivity.java



Etape 8 : Ajouter une « vue » en enfant de la ToolBar

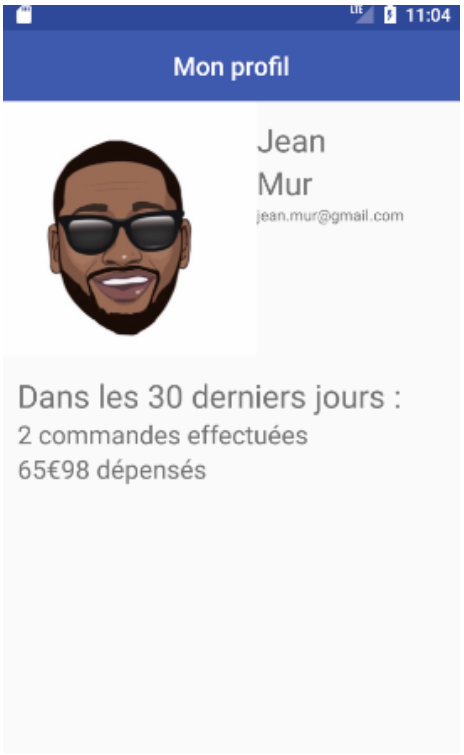
Comme on peut traiter une toolBar comme n'importe quel élément, on peut y ajouter simplement un élément en tant qu'enfant. Décommentez le TextView de la toolbar du fichier activity_profile.xml.

Passez comme argument false à la place de true à la méthode *setDisplayShowTitleEnabled* de ProfileActivity.java afin d'indiquer que l'on ne souhaite plus avoir le titre par défaut de l'application, mais notre propre titre représenté par le TextView.



Etape 9 : Customiser la ToolBar

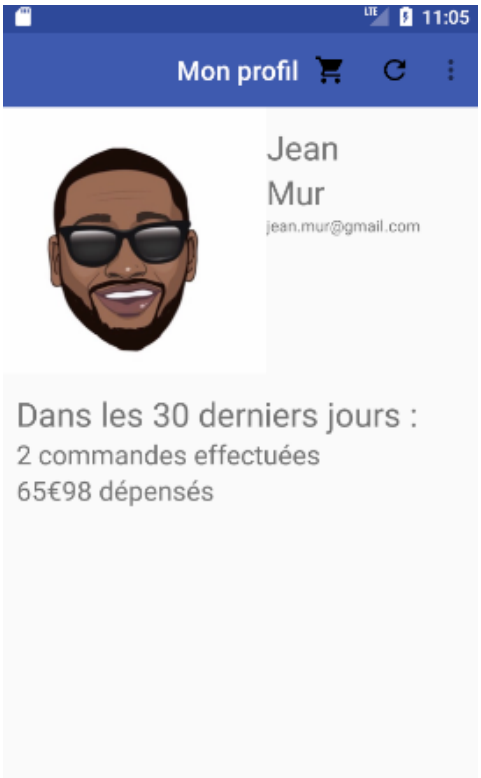
Il est donc maintenant très simple de customiser la ToolBar. Par exemple, pour centrer le titre, il suffit d'ajouter la ligne *android:layout_gravity="center"* au TextView de la toolbar du fichier *activity_profile.xml.* Faites le.



Etape 10 : Ajouter des actions principales et secondaires à une ToolBar

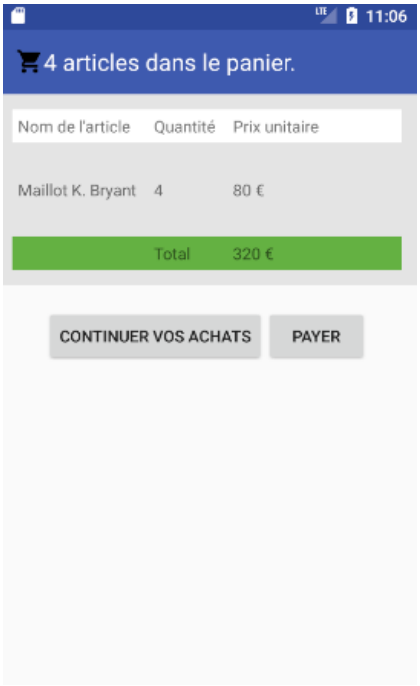
Même si on utilise maintenant une toolBar, nous pouvons toujours y ajouter un menu comme précédemment avec l'actionBar :

Décommentez les méthodes « *onCreateOptionsMenu* » et « *onOptionsItemSelected* » du fichier *ProfileActivity.java*



Etape 11 : Créer une seconde ToolBar

Dans une nouvelle activité, celle du panier, On peut créer une toolBar de la même façon :
Décommentez la déclaration de la toolbar dans le fichier activity_cart.xml
Décommentez l'affectation du texte du compteur dans le fichier CartActivity.java



Etape 12 : Extraire la déclaration d'une ToolBar et l'inclure dans une activité

Cependant, si on souhaite réutiliser une toolBar, on peut extraire la déclaration de celle-ci en la plaçant dans un fichier du layout. Ici, *custom_toolbar.xml.* Et inclure cette définition dans le layout de l'activité.

Supprimez (ou commenter) la déclaration de la « *customToolbar* » dans le fichier activity_cart.xml.

Décommentez la balise include de ce fichier.

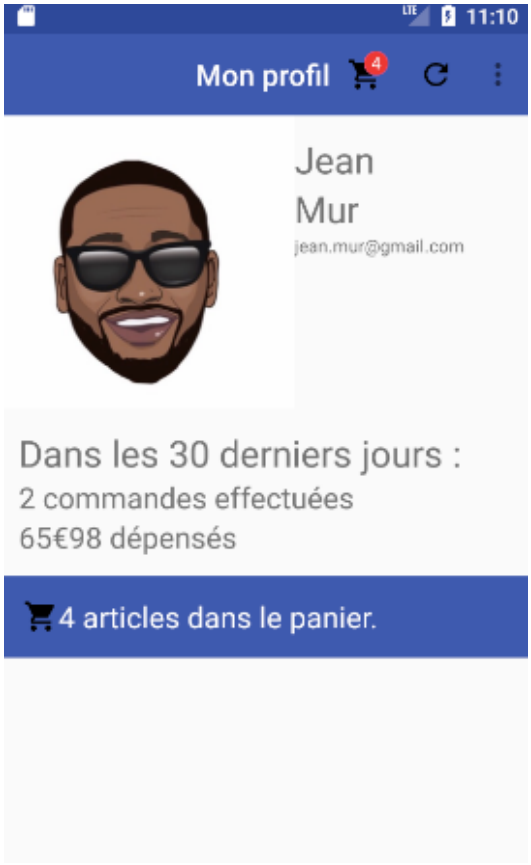
Vous pouvez voir que le résultat est le même qu'à l'étape précédente.

Etape 13 : Déclarer plusieurs ToolBar pour une activité

Enfin, un des avantages de la ToolBar est que l'on peut en déclarer plusieurs dans une même activité. On peut alors inclure la toolbar extraite précédemment dans l'activité du profil : ajoutez cette balise include à la fin du layout *d'activity_profile.xml*.

Décommentez l'affectation du texte du compteur dans le fichier *ProfileActivity.java*.

Vous pouvez constater que, comme la toolbar est un composant du layout, Elle subit les mêmes règles que les autres éléments du layout : à la suite des composant dans un linearLayout par exemple.



L'utilisation d'une toolbar permet donc une plus grande liberté sur son placement.

La toolBar permet aussi une plus grande liberté de customisation ainsi que la possibilité d'en mettre plusieurs dans une seule et même activité. De plus, la toolBar peut posséder les mêmes fonctionnalités que l'actionBar.

Dans les applications d'aujourd'hui, il est donc plus intéressant d'utiliser les toolBar, même si, dans certains cas basiques, l'actionBar peut être suffisante.

Informations complémentaires

https://guides.codepath.com/android/Using-the-App-Toolbar
-> tutoriel pour débutant sur l'utilisation des ToolBar

https://developer.android.com/reference/android/support/v7/widget/Toolbar.html
-> Documentation de ToolBar

Attention au piège :

Pour l'attribut showAsAction :

- Si l'activité étends « AppCompatActivity » alors il faut utiliser un namespace dans le fichier xml de définition du menu, exemple : « xmlns:app="http://schemas.android.com/apk/res-auto" » et app:showAsAction=« ifRoom »
- Si l'activité étends Activity alors pas besoin d'utiliser un nouveau namespace.
  —> android:showAsAction=« ifRoom »