

Le développement tablette Android



Sommaire

01

Spécificités

02

Optimiser les layouts

03

Optimiser les ressources

04

Distribuer une application

1

Spécificités

Déclarer les fonctions liées au matériel

Les smartphones et les tablettes offrent souvent des différences au niveau du matériel (ex : beaucoup de tablettes ne possèdent que le Wi-Fi)

Lors de la distribution de vos applications sur le Store assurez-vous que votre application ne nécessite pas l'utilisation de matériel indisponible sur les tablettes.

Exemple : présence du hardware nécessaire à la téléphonie sur l'appareil obligatoire ou pas pour le bon fonctionnement de votre application

```
<uses-feature android:name="android.hardware.telephony"
android:required="false" />
```

Il est possible de spécifier les tailles d'écran utilisées par l'application sur une tablette via l'élément <supports-screens>



2

Optimiser les layouts

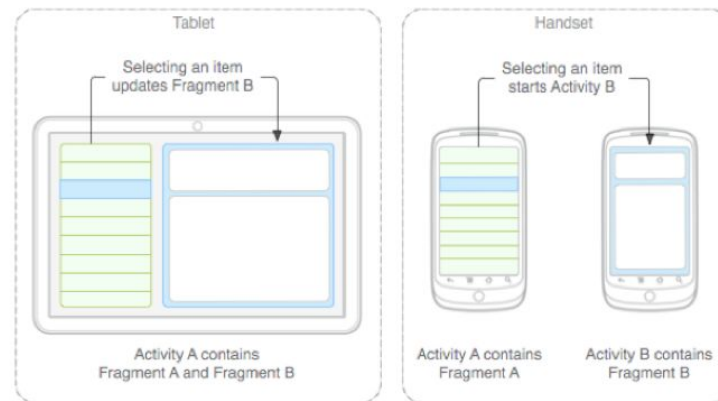
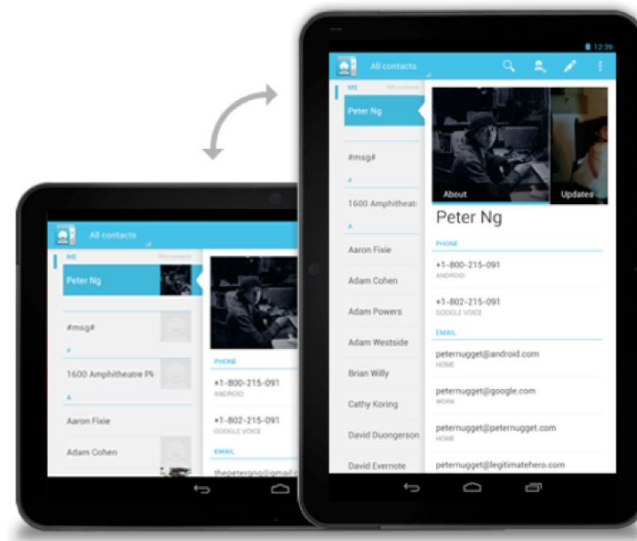
Objectif de l'optimisation

Utilisation de tout l'espace offert sur une tablette

- Composant le plus approprié : **Fragment**

Rappel sur les fragments:

- Permet un agencement plus simple des écrans complexes sur des résolutions plus importantes (tablettes).
- Permet de rassembler plusieurs morceaux d'interface dans une même activité par exemple.



Optimiser l'affichage sur tablette


Pour optimiser l'affichage sur tablette nous allons utiliser le concept des fragments et des “**qualifiers**”.


On crée un répertoire à la racine de res/ qui sera nommé layout-sw600dp. On y met uniquement les layout des écrans qu'on veut gérer sur tablette et on adapte l'affichage de ces écrans pour qu'ils soient conformes à l'affichage qu'on veut sur une tablette

Le **qualifier** -sw600dp correspond à “smallest width” de 600dp.


Android se charge de pointer vers ce répertoire gérant les layouts, pour les terminaux dont la largeur (peu importe l'orientation) fait au moins 600 dp


▼  res

>  drawable


▼  layout


▼  activity_login (2)

 activity_login.xml

 activity_login.xml (sw600dp)

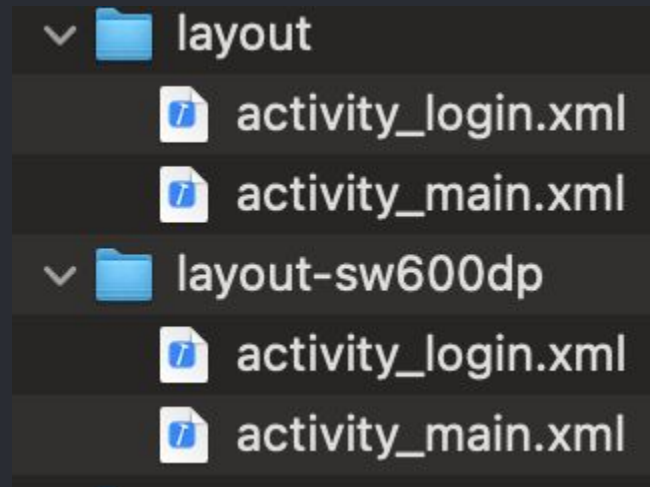
▼  activity_main (2)

 activity_main.xml

 activity_main.xml (sw600dp)

Optimiser l'affichage sur tablette

Vue des dossiers





Les qualifieurs

Smallest width: largeur minimale disponible (peu importe l'orientation)

Exemples:

- sw320dp
- sw600dp
- sw720dp

Quelques valeurs que vous pourriez utiliser selon les cas:

320, pour des appareils correspondants à:

- 240x320 ldpi (smartphone basse densité)
- 320x480 mdpi (smartphone)
- 480x800 hdpi (smartphone haute densité)

480, pour des appareils correspondants à:

- 480x800 mdpi (tablette/smartphone de moyenne densité).

600, pour des appareils correspondants à:

- 600x1024 mdpi (tablette 7")

720, pour des appareils correspondants à:

- 720x1280 mdpi (tablette 10")





Les qualifieurs

Available width: largeur minimale disponible (change selon l'orientation)

Exemples:

- w720dp
- w1024dp

Utile par exemple pour passer en mode liste / détail sur une même vue lors d'un passage en mode paysage





Les qualifieurs

Available height: hauteur minimale disponible (change selon l'orientation)

Exemples:

- h720dp
- h1024dp



Les qualifieurs

Screen size: taille de l'écran sur base de valeurs prédéfinies

Exemples:

- **small:**
La taille de mise en page minimale pour un petit écran est d'environ 320x426 unités dp.
- **normal:**
La taille de mise en page minimale pour un écran normal est d'environ 320x470 unités dp.
- **large:**
La taille de mise en page minimale pour un grand écran est d'environ 480x640 unités dp.
- **xlarge:**
La taille de mise en page minimale pour un écran xlarge est d'environ 720 x 960 unités dp. Dans la plupart des cas, les appareils avec des écrans extra-larges seraient très probablement des appareils de type tablette.





Les qualifieurs

Orientation: en fonction de l'orientation (portrait, paysage)

Exemples:

- **port:**
Ressource à utiliser en mode portrait.
- **land:**
Ressource à utiliser en mode landscape (paysage).





Les qualifieurs

Avantage de cette méthode : on peut gérer l'affichage sur smartphone également.

Désavantage de cette méthode : on ne peut pas personnaliser l'affichage tablette.

Remarque :

- Vous pouvez définir plusieurs **qualifieurs** afin d'affiner au mieux le choix de la ressource par le système
- Vous pouvez aussi les combiner
Ex: layout-port-hdpi





Les qualifieurs

Night mode: en fonction de l'état du night mode (actif ou pas)

Exemples:

- **night:**
Ressource à utiliser en night mode.
- **notnight:**
Ressource à utiliser en mode normal.





Les qualifieurs

Screen pixel density: en fonction de la densité en pixels de l'écran

Exemples:

- ldpi
- mdpi
- hdpi
- xhdpi
- xxhdpi
- xxxhdpi



4

Optimiser les ressources



Optimiser les ressources

Lors d'un module précédent, nous avons vu comment centraliser les différentes tailles afin de les réutiliser dans nos layouts (marges, tailles de texte...)

Comme pour les layouts, il est possible de dériver le fichier `dimens` afin d'améliorer le support de différentes tailles d'écran

Par-exemple, la taille des textes que vous utilisez pour vos titre pourrait sembler petite sur une tablette extra large

`values/dimens.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <dimen name="title_text_size">14sp</dimen>
</resources>
```

`values-xlarge/dimens.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <dimen name="title_text_size">48sp</dimen>
</resources>
```

Optimiser les ressources

Cela peut aussi être utile si votre application contient des ressources de type vidéo

Dans ce cas nous créerons les variantes de ces ressources dans les dossier raw

Ex:
raw/intro.mp4
raw-xlarge/intro.mp4
...

▼ raw

▼ intro (2)

intro.mp4

intro.mp4 (xlarge)

5

Distribuer une application



Distribuer vers des écrans en particulier

Il est recommandé de faire une application "universelle" afin de capter le maximum d'utilisateurs avec une seule application.

Cependant il existe certains cas où il est préférable de restreindre son application qu'à certains types de devices ou alors créer par exemple deux APK (1 smartphone + 1 tablette).

Propriétés <compatible-screens> et <supports-screens> pour spécifier les différents écrans supportés. Ces propriétés sont prises en compte par les services Google Play pour filtrer les applications lors des téléchargements ou exécuter votre application en mode compatibilité.



Support Screens

Il est possible de spécifier les tailles d'écran supportées par l'application via l'élément `<supports-screens>`

Ceci se configure dans votre fichier Manifest.

Attention, dans le cas où votre application est exécutée sur un appareil ne répondant pas aux critères définis.

Par-exemple si votre application supporte uniquement les petits écrans, le mode compatibilité sera activé dès lors qu'elle sera exécutée sur un appareil plus grand. Cela produira un effet de zoom ! **À éviter**

Aujourd'hui, il est plutôt conseillé de construire des layouts adaptatifs avec notamment les Constraint Layout

Cependant dans certains cas, l'application voudrait exclure certaines tailles de device, ou au contraire s'adapter différemment en fonction de différentes tailles

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.technipixl.qualifiers">

    <supports-screens
        android:resizeable="false"
        android:smallScreens="false"
        android:normalScreens="false"
        android:largeScreens="false"
        android:xlargeScreens="true"
        android:requiresSmallestWidthDp="600"/>

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Qualifiers"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Qualifiers">
```

Compatible Screens

Il est possible de spécifier les tailles d'écran compatibles avec l'application via l'élément `<compatible-screens>`

Ceci se configure dans votre fichier Manifest.

Les appareils ne correspondant pas aux critères définis n'auront pas accès à l'application sur le Google Play Store

android:screenSize:

- small
- normal
- large
- xlarge

android:screenDensity:

- ldpi (+- 120 dpi)
- mdpi (+-160 dpi)
- hdpi (+- 240 dpi)
- xhdp" (+- 320 dpi)
- 280
- 360
- 420
- 480
- 560

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

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
    package="com.technipixl.qualifiers">
```

```
    <compatible-screens>
```

```
        <!-- all normal size screens -->
```

```
        <screen android:screenSize="normal" android:screenDensity="ldpi" />
```

```
        <screen android:screenSize="normal" android:screenDensity="mdpi" />
```

```
        <screen android:screenSize="normal" android:screenDensity="hdpi" />
```

```
        <screen android:screenSize="normal" android:screenDensity="xhdpi" />
```

```
        <screen android:screenSize="normal" android:screenDensity="xxhdpi" />
```

```
        <screen android:screenSize="normal" android:screenDensity="xxxhdpi" />
```

```
        <!-- all large size screens -->
```

```
        <screen android:screenSize="large" android:screenDensity="ldpi" />
```

```
        <screen android:screenSize="large" android:screenDensity="mdpi" />
```

```
        <screen android:screenSize="large" android:screenDensity="hdpi" />
```

```
        <screen android:screenSize="large" android:screenDensity="xhdpi" />
```

```
        <screen android:screenSize="large" android:screenDensity="xxhdpi" />
```

```
        <screen android:screenSize="large" android:screenDensity="xxxhdpi" />
```

```
        <!-- all extra large size screens -->
```

```
        <screen android:screenSize="xlarge" android:screenDensity="ldpi" />
```

```
        <screen android:screenSize="xlarge" android:screenDensity="mdpi" />
```

```
        <screen android:screenSize="xlarge" android:screenDensity="hdpi" />
```

```
        <screen android:screenSize="xlarge" android:screenDensity="xhdpi" />
```

```
        <screen android:screenSize="xlarge" android:screenDensity="xxhdpi" />
```

```
        <screen android:screenSize="xlarge" android:screenDensity="xxxhdpi" />
```

```
    </compatible-screens>
```

APK et App Bundle

Il est possible de publier deux types d'artefacts sur le Google Play Store

- APK
- AAB (App Bundle)

L'App Bundle a un avantage par-rapport à l'APK

En effet un APK contiendra toutes les ressources, y compris celles inutilisées lors de la distribution sur un appareil

À l'inverse, en utilisant un App Bundle, le Google Play Store sera en mesure de générer un APK sur base des caractéristiques de l'appareil cible

Celui-ci ne contiendra que les ressources nécessaires

- Diminution de la taille lors du téléchargement
- Recommandé par Google

Un App Bundle pour les gouverner tous...



ex1

Exercise



Implémenter une vue liste - détail

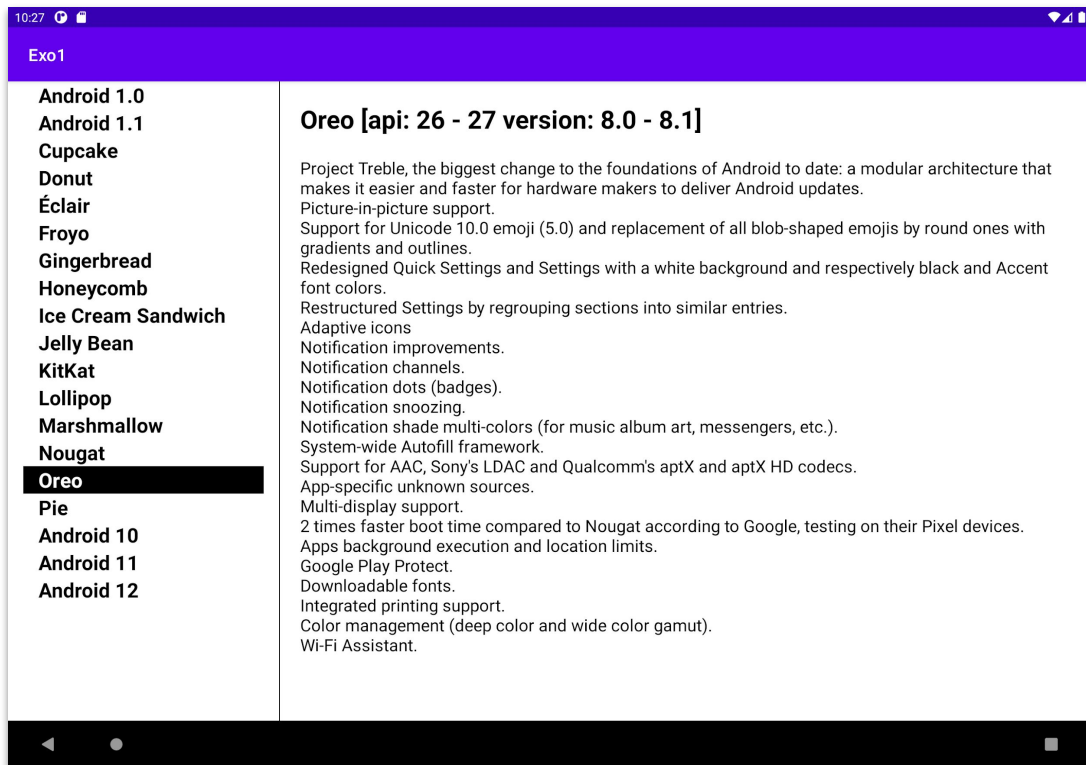
Nous allons mettre en pratique ce que nous avons appris dans ce module afin d'implémenter une activité qui devra gérer un mode liste et détail.

Sur un **smartphone** nous afficherons la liste sur un premier écran. Suite au clic sur un élément de la liste, nous afficherons le détail dans un nouvel écran.

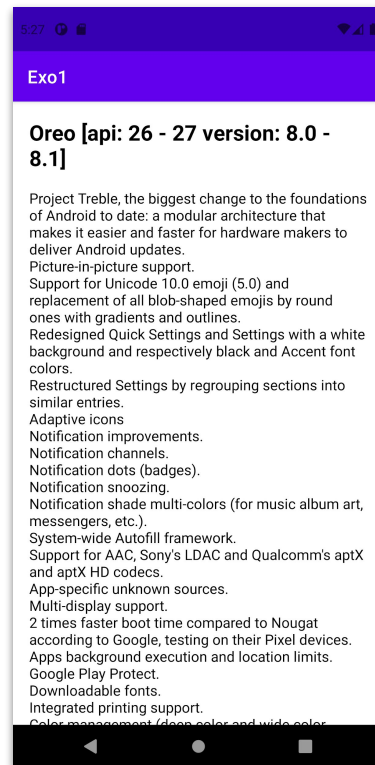
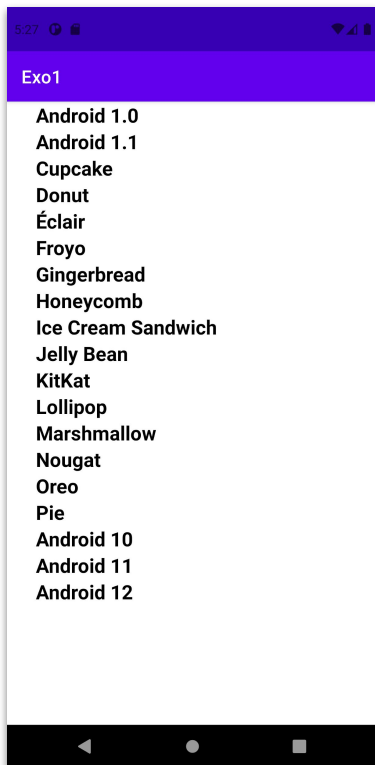
Sur **tablette** nous afficherons la liste et le détail dans le même écran. Nous modifierons le détail suite au clic sur un élément de la liste.



Implémenter une vue liste - détail



Implémenter une vue liste - détail





Avez-vous des questions?

neopixl.

A SMILE GROUP COMPANY



Restons en contact

neopixl.

A SMILE GROUP COMPANY

115A, Rue Emile Mark
L-4620 Differdange

(+352) 26 58 06 03
contact@neopixl.com