CESI

Formation Android

Stéphane Castrec

Ingénieur logiciel
- Développeur au CM Arkéa
- CTO à EQwall

@_stephane_

StephaneC

Objectifs de l'atelier

Maitriser et être autonome sur les éléments suivants:

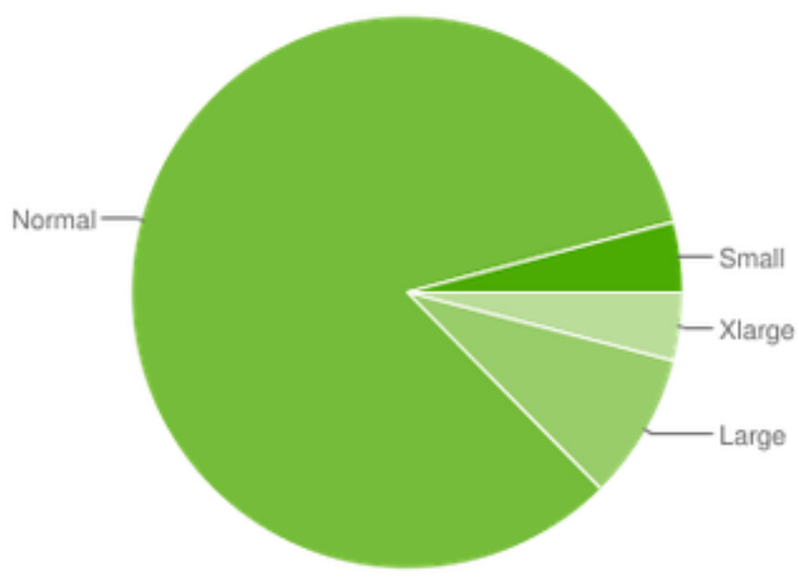
- Comprendre la fragmentation
- <u>Savoir utiliser les fragments</u>

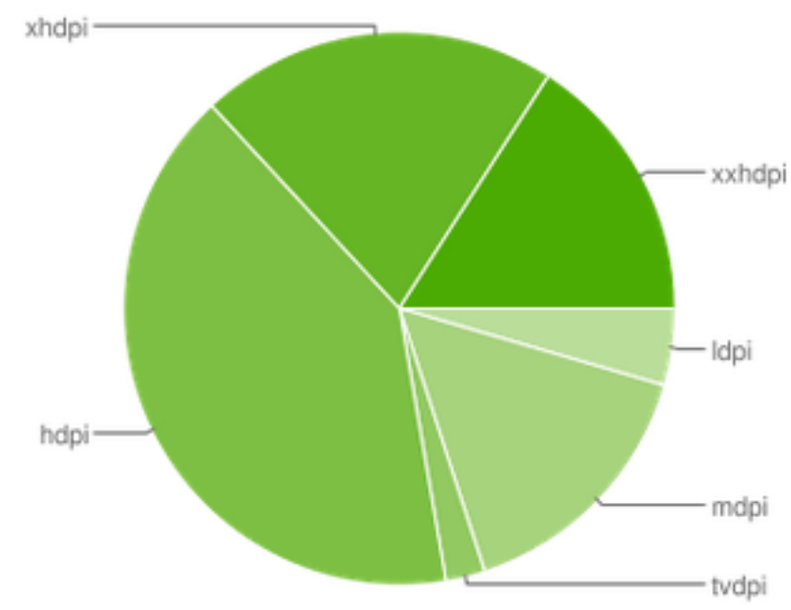
Android Fragmentation

- Taille écran
- Densité de l’écran
- Version de l’OS

Screen sizes

	ldpi	mdpi	tvdpi	hdpi	xhdpi	xxhdpi	**Total**
Small	4.1%						4.1%
Normal		7.6%	0.1%	39.9%	19.8%	15.9%	83.3%
Large	0.4%	4.8%	2.2%	0.6%	0.6%		8.6%
Xlarge		3.1%		0.3%	0.6%		4.0%
Total	4.5%	15.5%	2.3%	40.8%	21.0%	15.9%	





Data collected during a 7-day period ending on June 1, 2015.

Any screen configurations with less than 0.1% distribution are not shown.

https://developer.android.com/about/dashboards/index.html?utm_source=suzunone

Vue intéressante

Device metrics

Type	Device	Platform	Screen dimensions in cm		Aspect Ratio	Width × Height dp	Width × Height px	Density
	Android One	Android	4.5 in	2.2 × 3.9 in	16 : 9	320 × 569 dp	480 × 854 px	1.5 hdpi
	Asus Zen Watch	Android	1.6 in	1.2 × 1.2 in	1 : 1	213 × 213 dp	320 × 320 px	1.5 hdpi
	Dell Venue 8	Android	8.4 in	4.5 × 7.1 in	16 : 10	800 × 1280 dp	1600 × 2560 px	2.0 xhdpi
	HTC One M8	Android	5.0 in	2.5 × 4.4 in	16 : 9	360 × 640 dp	1080 × 1920 px	3.0 xxhdpi
	HTC One M9	Android	5.0 in	2.5 × 4.4 in	16 : 9	360 × 640 dp	1080 × 1920 px	3.0 xxhdpi
	LG G Watch	Android	1.7 in	1.2 × 1.2 in	1 : 1	187 × 187 dp	280 × 280 px	1.5 hdpi
	LG G Watch R	Android	1.8 in	1.3 × 1.3 in	1 : 1	213 × 213 dp	320 × 320 px	1.5 hdpi
	LG G2	Android	5.2 in	2.5 × 4.5 in	16 : 9	360 × 640 dp	1080 × 1920 px	3.0 xxhdpi
	LG G3	Android	5.5 in	2.7 × 4.8 in	16 : 9	480 × 853 dp	1440 × 2560 px	3.0 xxhdpi
	Moto 360	Android	1.6 in	1.6 × 1.4 in	32 : 29	241 × 218 dp	320 × 290 px	1.3 tvdpi
	Moto G	Android	4.5 in	2.2 × 3.9 in	16 : 9	360 × 640 dp	720 × 1280 px	2.0 xhdpi

Keynote Fichier Édition Insertion Diapositive Format Disposition Présentation Lecture Partager Fenêtre Aide 100 %

http://www.google.com/design/tool/devices/

Gérer la densité

Gérer la densité

- Ne “*jamais*” utiliser de pixels

—> utilisation des dip

- Différents images pour différentes densités

- `xhdpi` : 2.0
- `hdpi` : 1.5
- `mdpi` : 1.0 (baseline)
- `ldpi` : 0.75

```
MyProject/
  res/
    drawable-xhdpi/
        awesomeimage.png
    drawable-hdpi/
        awesomeimage.png
    drawable-mdpi/
        awesomeimage.png
    drawable-ldpi/
        awesomeimage.png
```

```
res/...
    mipmap-ldpi/...
        finished_launcher_asset.png
    mipmap-mdpi/...
        finished_launcher_asset.png
    mipmap-hdpi/...
        finished_launcher_asset.png
    mipmap-xhdpi/...
        finished_launcher_asset.png
    mipmap-xxhdpi/...
        finished_launcher_asset.png
    mipmap-xxxhdpi/...
        finished_launcher_asset.png
```

Gérer les tailles

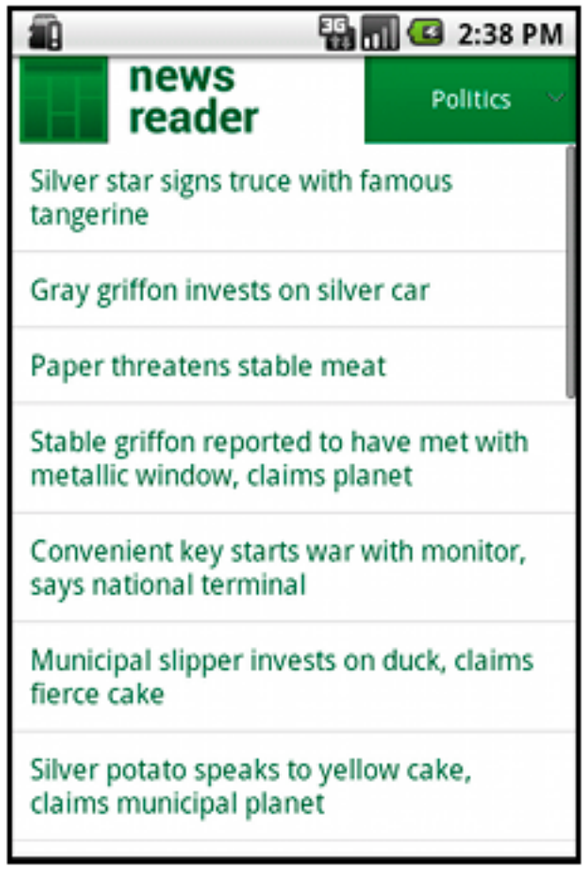
D’abord les layouts

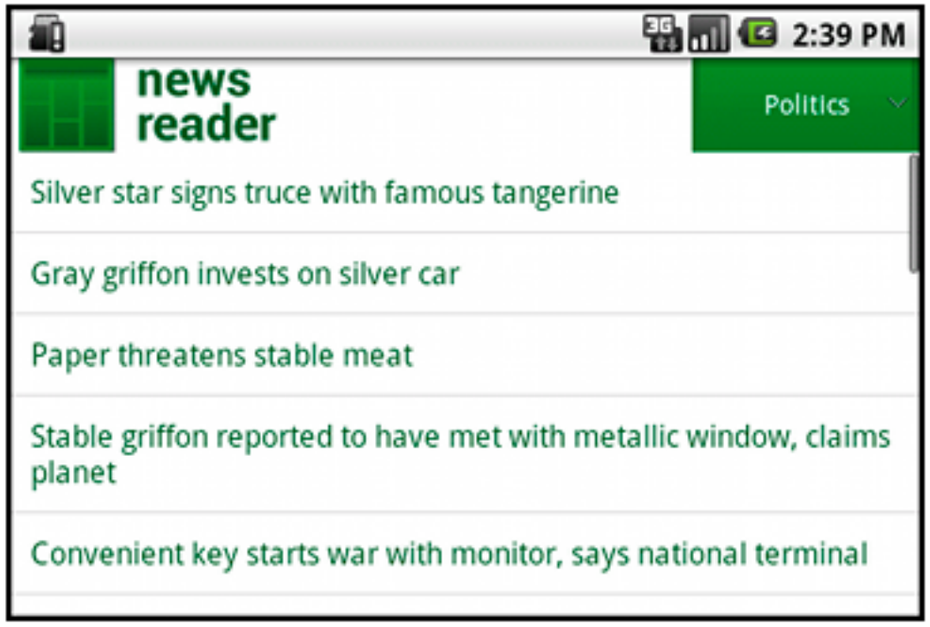
D’abord les layouts

- Taille des layouts
  - match_parent / wrap_content
- RelativeLayout
- 9patch
- Alias & les qualifier

D’abord les layouts

Les tailles





D’abord les layouts

RelativeLayout


```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <TextView
        android:id="@+id/label"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Type here:"/>
    <EditText
        android:id="@+id/entry"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/label"/>
    <Button
        android:id="@+id/ok"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/entry"
        android:layout_alignParentRight="true"
        android:layout_marginLeft="10dp"
        android:text="OK" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_toLeftOf="@id/ok"
        android:layout_alignTop="@id/ok"
        android:text="Cancel" />
</RelativeLayout>
```

D’abord les layouts
Les qualifiers

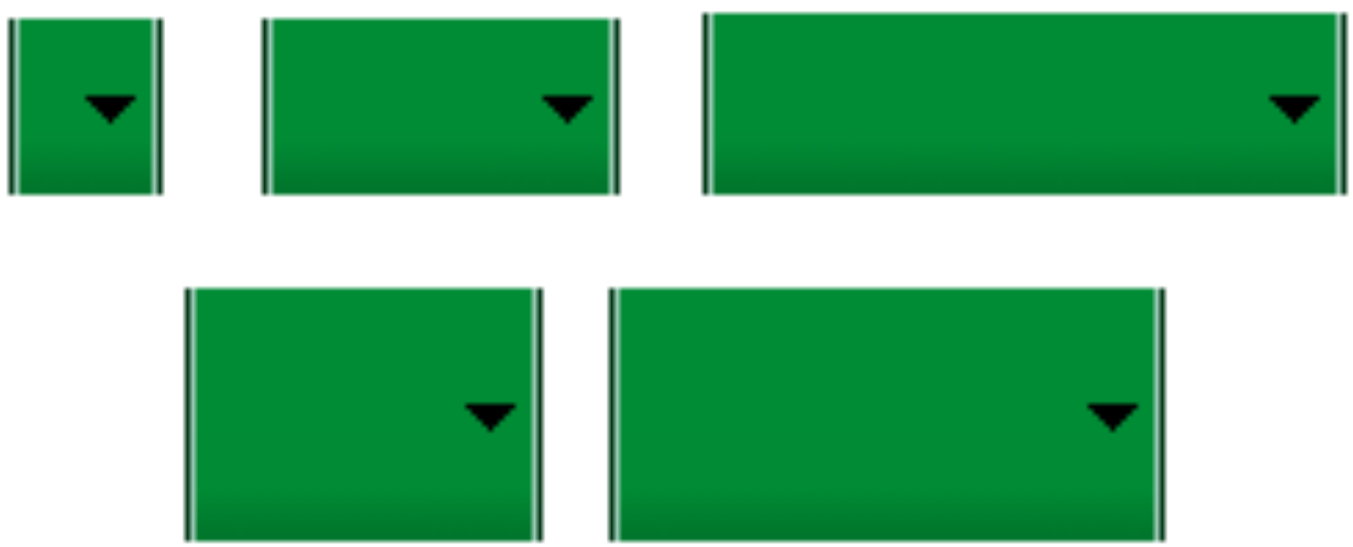
Objectifs: utiliser différents layout/*.xml en fonction du média

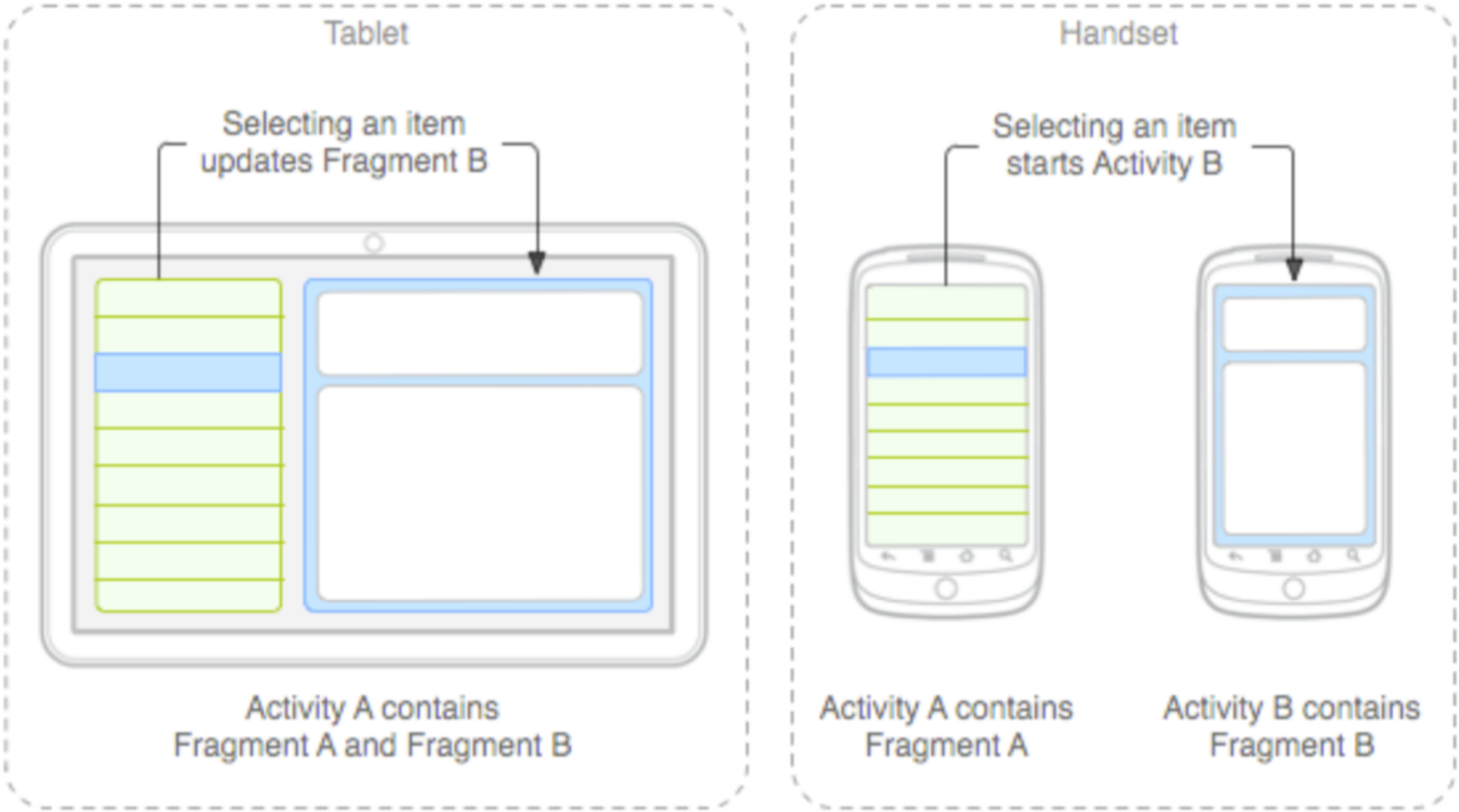
```
res/layout/my_layout.xml                // layout for normal screen size ("default")
res/layout-large/my_layout.xml          // layout for large screen size
res/layout-xlarge/my_layout.xml         // layout for extra-large screen size
res/layout-xlarge-land/my_layout.xml    // layout for extra-large in landscape orientation

res/drawable-mdpi/graphic.png           // bitmap for medium-density
res/drawable-hdpi/graphic.png           // bitmap for high-density
res/drawable-xhdpi/graphic.png          // bitmap for extra-high-density
res/drawable-xxhdpi/graphic.png         // bitmap for extra-extra-high-density

res/mipmap-mdpi/my_icon.png           // launcher icon for medium-density
res/mipmap-hdpi/my_icon.png           // launcher icon for high-density
res/mipmap-xhdpi/my_icon.png          // launcher icon for extra-high-density
res/mipmap-xxhdpi/my_icon.png         // launcher icon for extra-extra-high-density
res/mipmap-xxxhdpi/my_icon.png        // launcher icon for extra-extra-extra-high-density
```

9 Patch




Tablet
Selecting an item updates Fragment B
Activity A contains Fragment A and Fragment B
Handset
Selecting an item starts Activity B
Activity A contains Fragment A
Activity B contains Fragment B

Guidelines

Comment faire simple?

Les fragments

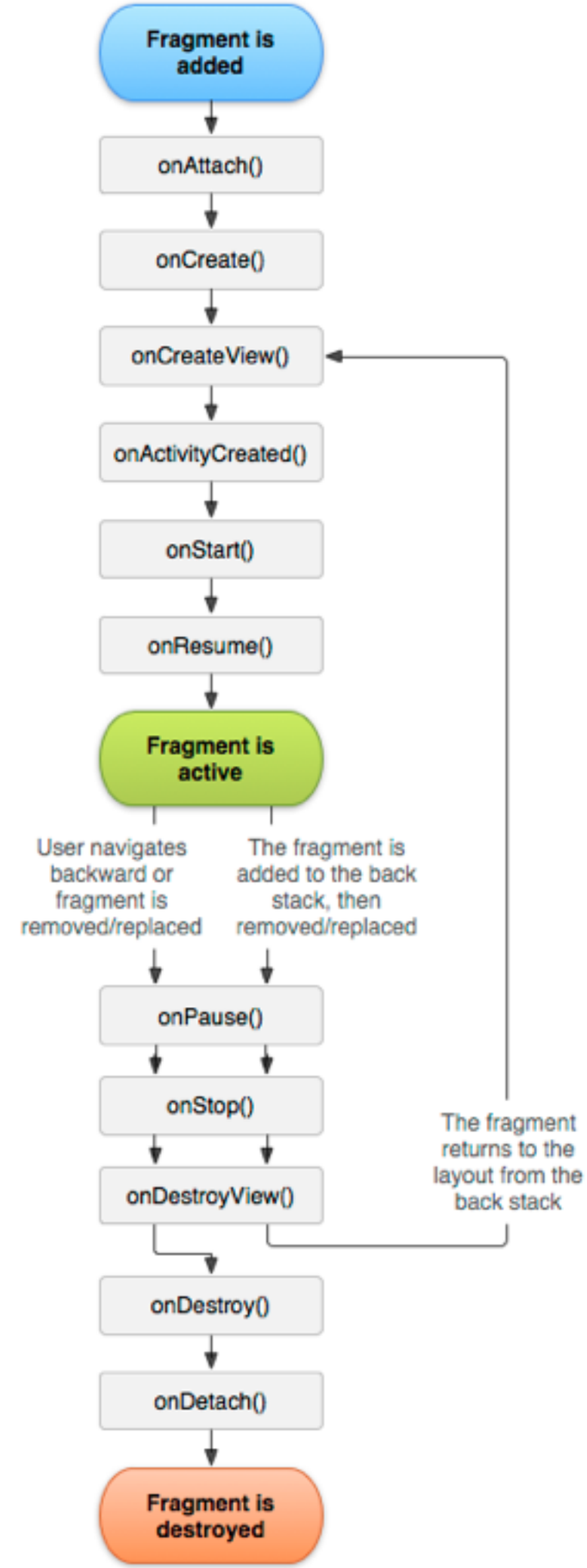
http://proverbspurple.files.wordpress.com/2011/05/jigsaw-puzzle.jpg

Fragments

Découpage en composants fonctionnels,
indépendants et réutilisables

Fragments

- Widget
  - Sous partie d'une activité
- Cycle de vie dédié
- API Level 11
  - Android lib support v4



http://developer.android.com/guide/components/fragments

Fragments

- Un fichier layout dédié
- Extends Fragment
- Override onCreateView
- Dans le xml, utiliser <Fragment class=« »/>

http://developer.android.com/guide/components/fragments.html

Exemple Java

```java
public static class ExampleFragment extends Fragment {
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.example_fragment, container, false);
    }
}
```

http://developer.android.com/guide/components/fragments.html

Utilisation

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <fragment android:name="com.example.news.ArticleListFragment"
            android:id="@+id/list"
            android:layout_weight="1"
            android:layout_width="0dp"
            android:layout_height="match_parent" />
    <fragment android:name="com.example.news.ArticleReaderFragment"
            android:id="@+id/viewer"
            android:layout_weight="2"
            android:layout_width="0dp"
            android:layout_height="match_parent" />
</LinearLayout>
```

http://developer.android.com/guide/components/fragments.html

Communication inter-fragments

Ne sera pas implémentée dans les ateliers

Communication inter-fragments

Activity sert de proxy

- fragmentA -> activity -> fragmentB
- Version officielle

Event bus

- Version officieuse
- Plus simple mais utilisation de lib tierce
- Perte en lisibilité

Communication inter-fragments: Version officielle

1 : le fragment défini une callback

```
public static class FragmentA extends ListFragment {
    ...
    // Container Activity must implement this interface
    public interface OnArticleSelectedListener {
        public void onArticleSelected(Uri articleUri);
    }
    ...
}
```

Communication inter-fragments: Version officielle

2 : L’activity implémente la callback

Communication inter-fragments: Version officielle

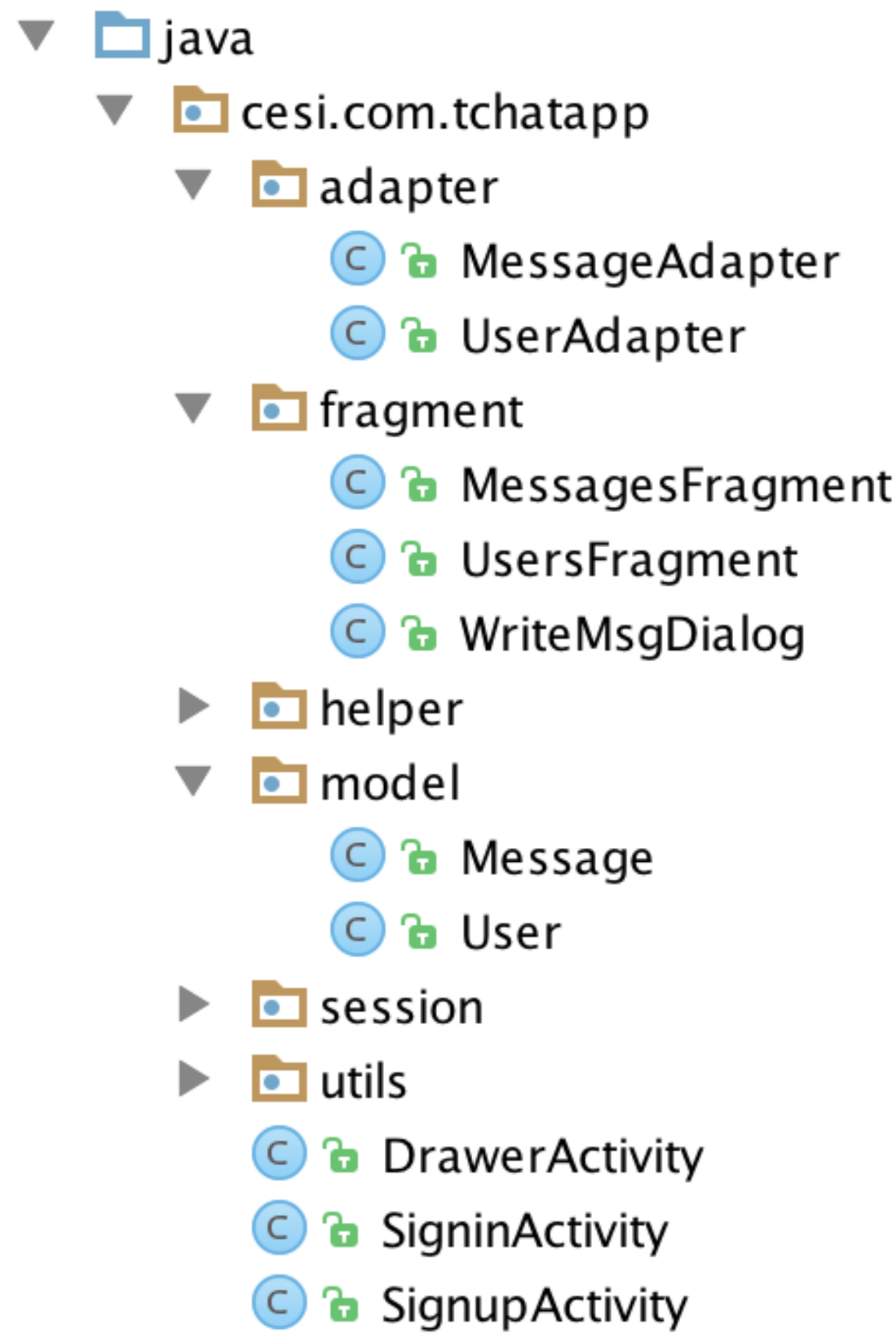
3 : Le fragment attache la callback à l'activity

```
public static class FragmentA extends ListFragment {
    OnArticleSelectedListener mListener;
    ...
    @Override
    public void onAttach(Activity activity) {
        super.onAttach(activity);
        try {
            mListener = (OnArticleSelectedListener) activity;
        } catch (ClassCastException e) {
            throw new ClassCastException(activity.toString() + " must implement OnArticl
        }
    }
    ...
}
```

Et donc maintenant notre code pourrait fonctionner aussi sur tablette?

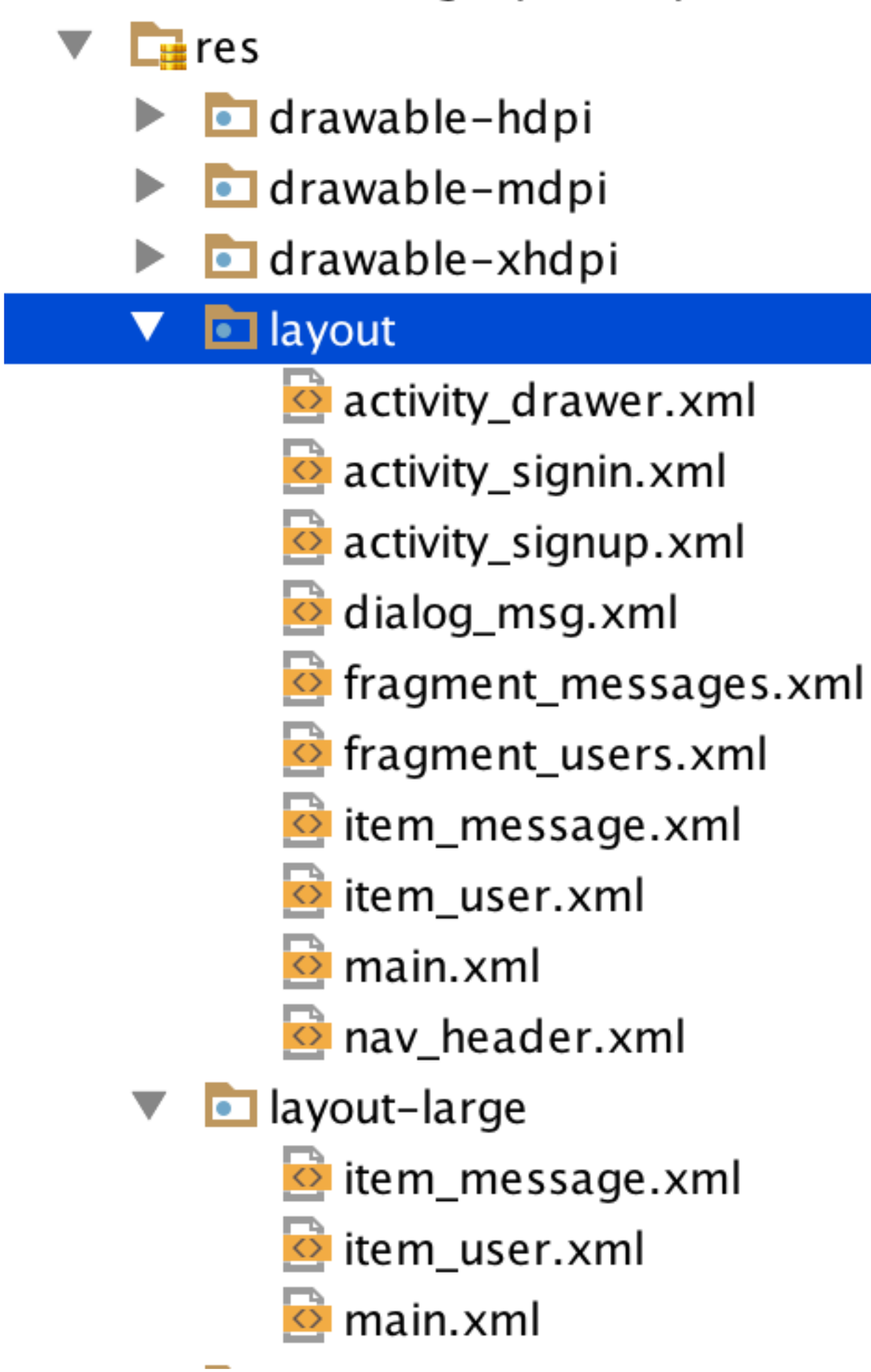

CHALLENGE ACCEPTED
LET'S CODE IT
memegenerator.net

Etape 1: passer notre app avec des fragments



Faire un point de synchro

Etape 2: Créer des layouts pour les tablettes



Etape 3: Utiliser les composants adéquates

```
if(!isLarge()) {
    mDrawerLayout = (DrawerLayout) findViewById(R.id.drawer_layout);
    ViewPager viewPager = (ViewPager) findViewById(R.id.viewpager);
    if (viewPager != null) {
        setupViewPager(viewPager);
    }
    TabLayout tabLayout = (TabLayout) findViewById(R.id.tabs);
    tabLayout.setupWithViewPager(viewPager);
}
```