



Eclipse – Android SDK

Les Fragments dynamiques



ANDROID

Introduction

- Les fragments statiques associent dans le layout un fragment à une classe de gestion du fragment.
- Avec les fragments dynamique cette association statique est omise. L'association est faite au niveau du code de l'activité en charge du fragment. Elle peut donc à tout moment remplacer un fragment par un autre.

Fragment statique

```
<LinearLayout>
  <fragment
    android:id="@+id/ListFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_marginTop="10dp"
    android:name=".FragmentBouton" >
  </fragment>
</LinearLayout>
```

Classe fragment

Fragment dynamique

```
<LinearLayout>
  <FrameLayout
    android:id="@+id/listeCapteursFragment"
    android:layout_width="0dp"
    android:layout_weight="1"
    android:layout_height="match_parent"
    android:layout_marginTop="10dp">
  </FrameLayout>
</LinearLayout>
```

Conteneur de
fragment dynamiques

```
FragmentManager fr = getSupportFragmentManager();
FragmentTransaction ft = fr.beginTransaction();
ListeCapteursFragment fgauche =
    new ListeCapteursFragment();
ft.add(R.id.listeCapteursFragment, fgauche);
ft.commit();
```



ANDROID

Exemple de Fragments dynamiques :

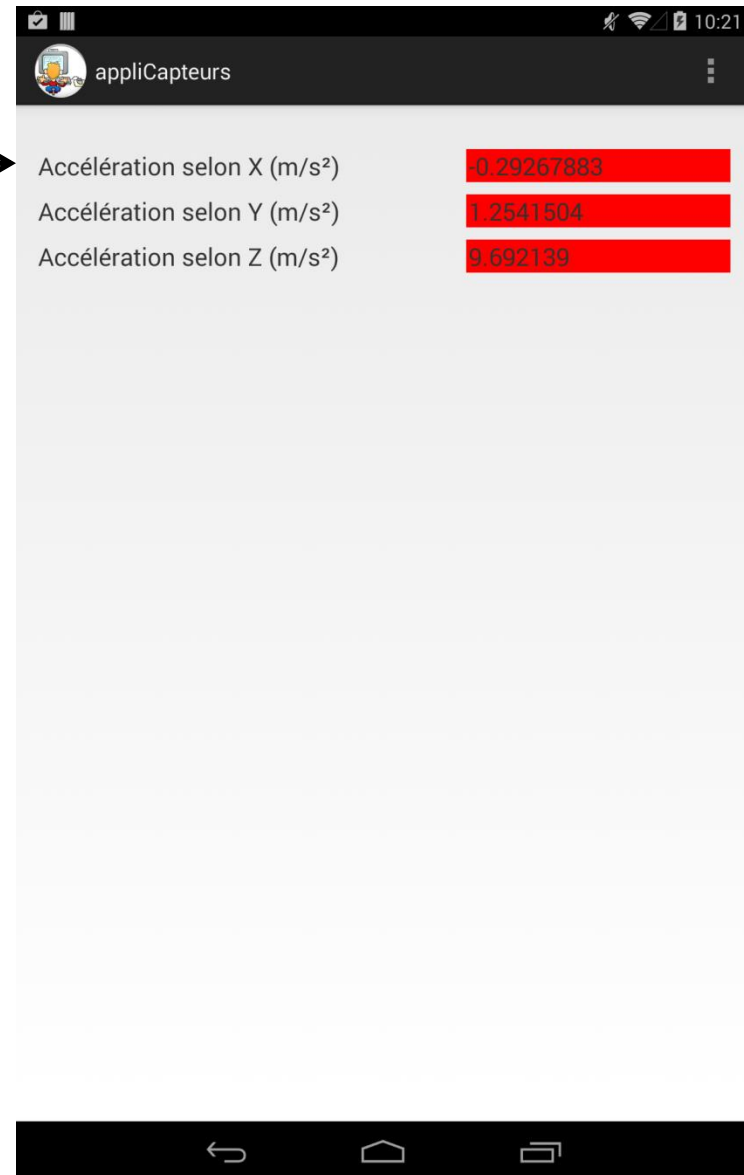
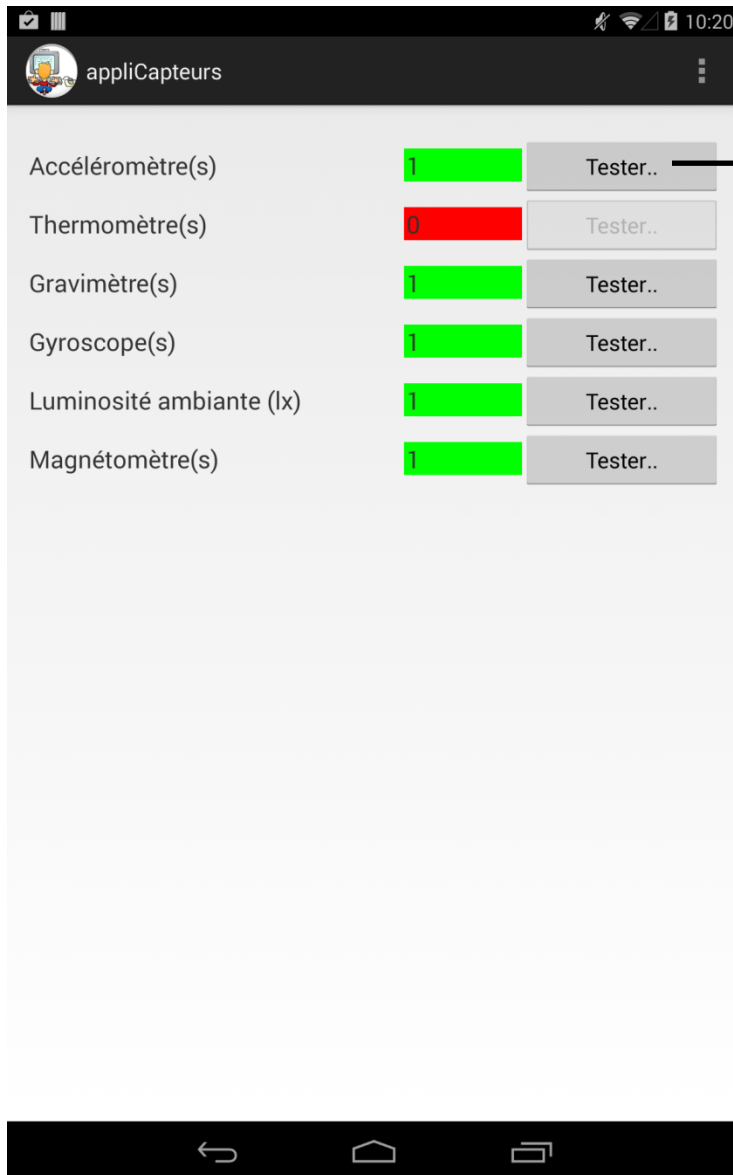
Une application qui teste les capteurs présents sur le device dans des fragments :

- ✓ *un fragment avec le nombre de capteurs présents par type de capteur,*
- ✓ *un fragment par type de capteur permettant de tester celui-ci en affichant périodiquement les valeurs mesurées.*

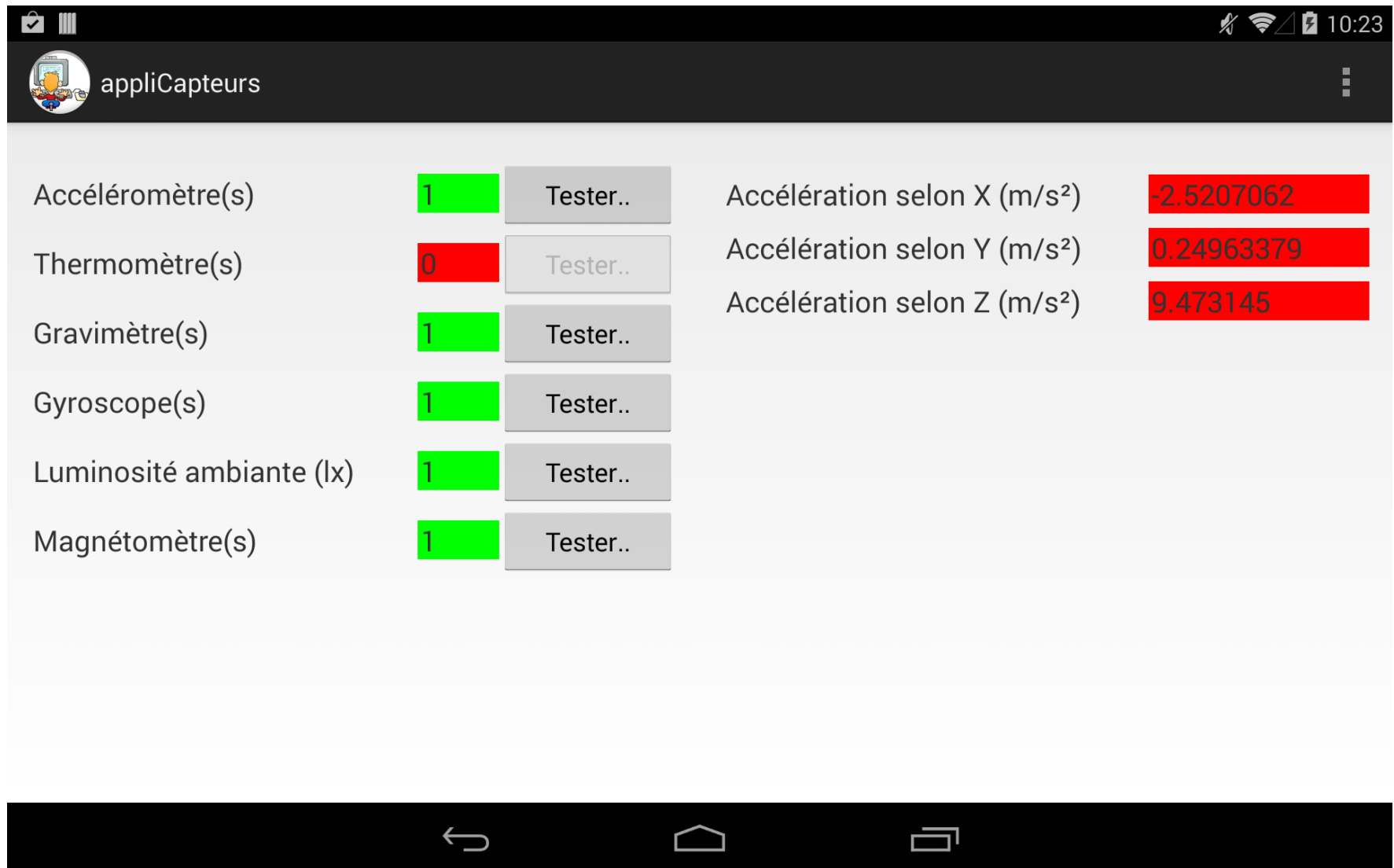
 En mode portrait, l'activité principale affiche le fragment présentant le nombre de capteurs présents par type. La sélection du test d'un capteur remplace le fragment courant par le fragment de test du capteur.

 En mode paysage l'activité principale affiche le fragment présentant le nombre de capteurs présents par type dans la moitié gauche de l'écran, la moitié droite affichant le fragment de test du capteur sélectionné ou un message d'invite si aucun capteur n'est sélectionné.

Le rendu attendu en mode portrait



Le rendu attendu en mode paysage



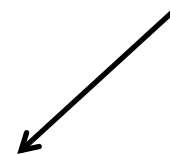


Le Layout de l'activité principale en mode portrait :

Activité principale portrait : *activity_main.xml*

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:baselineAligned="false"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="horizontal" >
    <FrameLayout
        android:id="@+id/listeCapteursFragment"
        android:layout_width="0dp"
        android:layout_weight="1"
        android:layout_height="match_parent"
        android:layout_marginTop="10dp">
    </FrameLayout>
</LinearLayout>
```

Le conteneur de
fragment dynamique





ANDROID

Le Layout de l'activité principale en mode paysage :

Activité principale paysage : *layout-land/activity_main.xml*

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:baselineAligned="false"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="horizontal" >
    <FrameLayout
        android:id="@+id/ListeCapteursFragment"
        android:layout_width="0dp"
        android:layout_weight="1"
        android:layout_height="match_parent"
        android:layout_marginTop="10dp">
    </FrameLayout>
    <FrameLayout
        android:id="@+id/detailCapteurFragment"
        android:layout_width="0dp"
        android:layout_weight="1"
        android:layout_height="match_parent"
        android:layout_marginTop="10dp">
    </FrameLayout>
</LinearLayout>
```

Les conteneurs
de fragments
dynamiques de
gauche et droite



ANDROID

Le Layout du fragment « liste » des capteurs présents :

fragment_listecapteurs.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android">
    <TableLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:stretchColumns="0,1,2">
        <TableRow>
            <TextView
                android:id="@+id/txtAccelerometer"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="@string/txtAccelerometer"
                android:layout_weight="0.5"
                android:textSize="20sp"/>
            <TextView
                android:id="@+id/valAccelerometer"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="@string/valAccelerometer"
                android:layout_weight="0.5"
                android:textSize="20sp"/>
            <Button
                android:id="@+id/btnAccelerometer"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="0.5"
                android:text="@string/to_Accelerometer"
                android:enabled="false"/>
        </TableRow>
    </TableLayout>
</LinearLayout>
```

etc, item pour les autres capteurs



ANDROID

Le Layout du fragment de test de l'accéléromètre :

fragment_accelerometer.xml

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <TableLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:stretchColumns="0,1,2">
        <TableRow
            android:layout_marginTop="10dip">
            <TextView
                android:id="@+id/txtAccX"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="@string/txtAccX"
                android:layout_weight="0.5"
                android:textSize="20sp"/>
            <TextView
                android:id="@+id/valAccX"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="@string/valAccX"
                android:layout_weight="0.5"
                android:textSize="20sp" />
        </TableRow>
    </TableLayout>
</LinearLayout>
```

**On a ainsi un fragment
par type de capteur
selon le même principe**

fragment_accelerometer.xml
fragment_temperature.xml
fragment_gravimeter.xml
fragment_gyroscopemeter.xml
fragment_luminometer.xml
fragment_magnetometer.xml

**Etc, item pour chaque
mesure**

Le Layout du message d'invite (fragment de droite en mode paysage) si aucun capteur sélectionné : **fragment_invite.xml**



כוכב

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_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" >

    <TextView
        android:id="@+id/txtDetailInitial"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAlignment="center"
        android:text="@string/txtDetailInitial"
        android:textSize="20sp"
    />
</LinearLayout>
```



La classe du fragment de test de l'accéléromètre :

```
public class AccelerometerFragment extends Fragment {
    private SensorManager manager;
    private TextView accX;
    private TextView accY;
    private TextView accZ;
```

Gestionnaire de capteurs

Chargement du fragment et création de la vue

@Override

```
public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
    View view = inflater.inflate(R.layout.fragment_accelerometer, container, false);
    this.accX=(TextView)view.findViewById(R.id.valAccX);
    this.accY=(TextView)view.findViewById(R.id.valAccY);
    this.accZ=(TextView)view.findViewById(R.id.valAccZ);
    manager = (SensorManager) getActivity().getSystemService(Context.SENSOR_SERVICE);
    manager.registerListener(mySensorEventListener, manager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER),
    SensorManager.SENSOR_DELAY_NORMAL);
    return view;
}
```

Initialisation du capteur accéléromètre

Classe interne, écouteur des événements accéléromètre

```
final SensorEventListener mySensorEventListener = new SensorEventListener()
```

```
{
    @Override
    public void onSensorChanged(SensorEvent sensorEvent) {
        {
            if(sensorEvent.sensor.getType()==Sensor.TYPE_ACCELEROMETER) {
                float xAcc=sensorEvent.values[0]; float yAcc=sensorEvent.values[1]; float zAcc=sensorEvent.values[2];
                accX.setText(String.valueOf(xAcc)); accY.setText(String.valueOf(yAcc)); accZ.setText(String.valueOf(zAcc));
            }
        }
    }
}
```

Déclenché à chaque nouvelle mesure

@Override

```
public void onAccuracyChanged(Sensor arg0, int arg1) {}
};
}
```

Surcharge obligatoire, non utilisé dans l'exemple

On retrouve une classe selon le même principe pour chaque capteur



La classe du fragment avec la « liste » des capteurs :

```
public class ListeCapteursFragment extends Fragment {

    private TextView valAccelerometers;
    private Button btnAcc;

    // Ecouter pour signaler
    private OnCapteurSelectedListener listenerChoixCapteur;

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
        Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.fragment_listecapteurs, container, false);
        // Init des widgets
        this.valAccelerometers=(TextView) view.findViewById(R.id.valAccelerometer);
        this.btnAcc=(Button) view.findViewById(R.id.btnAccelerometer);

        btnAcc.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                sendListenerCapteurChoice(Sensor.TYPE_ACCELEROMETER);
            }
        });
    }
}
```

Widgets pour la partie accéléromètre

L'interface écouteur des événements du fragment

Chargement du fragment et création de la vue

Clic sur le bouton de test du capteur

On retrouve le même principe de code pour chaque capteur



Suite...

```
// Test des capteurs
SensorManager manager =
    (SensorManager) getActivity().getSystemService(Context.SENSOR_SERVICE);

List<Sensor> listeAcc = manager.getSensorList(Sensor.TYPE_ACCELEROMETER);

if (listeAcc != null)
{
    this.valAccelerometers.setText(String.valueOf( listeAcc.size()));
    if(listeAcc.size() !=0)
    {
        this.btnAcc.setEnabled(true);
        this.valAccelerometers.setBackgroundColor(0xFF00FF00);
    }
}
else
    this.valAccelerometers.setText("0");
```

← Gestionnaire système des capteurs

← Obtenir la liste des capteurs de type accéléromètre

← Afficher le nombre de capteurs trouvés

On retrouve le même principe de code pour chaque capteur



Suite...

Attachement de l'activité conteneur comme
écouteur des événements du fragment

@Override

```
public void onAttach(Activity activity) {  
    super.onAttach(activity);  
    if (activity instanceof OnCapteurSelectedListener) {  
        this.listenerChoixCapteur = (OnCapteurSelectedListener) activity;  
    } else {  
        throw new ClassCastException(activity.toString()  
            + " Doit implémenter ListeCapteurFragment.OnCapteurSelectedListener");  
    }  
}
```

Interface écouteur

```
// Pour signaler à l'activité conteneur du fragment  
// la sélection d'un capteur dans le fragment  
// l'activité doit implémenter l'interface...
```

```
public interface OnCapteurSelectedListener {  
    public void onCapteurSelecteur(int sensorType);  
}
```

Signale à l'activité écouteur le capteur à tester

```
public void sendListenerCapteurChoice(int sensorType)  
{  
    this.listenerChoixCapteur.onCapteurSelecteur(sensorType);  
}
```

```
}
```



La classe MainActivity...

```

public class MainActivity extends FragmentActivity implements
ListeCapteursFragment.OnCapteurSelectedListener { ← Interface écouteur capteur
                                                    sélectionné

private final String FRAGLISTEPERSISTANT_KEY = "FRAGLISTEPERSISTANT_KEY";
private final String FRAGCAPTEURPERSISTANT_KEY = "FRAGCAPTEURPERSISTANT_KEY";
@Override
protected void onCreate(Bundle savedInstanceState) { ← Clés pour la gestion de la
                                                    persistance
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    FragmentManager fr = getSupportFragmentManager(); ← Gestionnaire de
                                                         fragments dynamiques
    FragmentTransaction ft = fr.beginTransaction();
    if(savedInstanceState == null) ← Premier chargement, pas d'état sauvegardé
    {
        ListeCapteursFragment fgauche = new ListeCapteursFragment();
        ft.add(R.id.listeCapteursFragment, fgauche); ← Chargement fragment liste
        ft.commit();
        if (getResources().getConfiguration().orientation ==
            Configuration.ORIENTATION_LANDSCAPE)
        {
            InviteFragment fdroite = new InviteFragment();
            ft = fr.beginTransaction(); ← Si mode paysage,
                                       chargement fragment
                                       d'invite
            ft.add(R.id.detailCapteurFragment, fdroite);
            ft.commit();
        }
    }
}

```

```

else ←
{
String fragListeName = savedInstanceState.getString(this.FRAGLISTEPERSISTANT_KEY);
String fragCapteurName = savedInstanceState.getString(this.FRAGCAPTEURPERSISTANT_KEY);
if (getResources().getConfiguration().orientation == Configuration.ORIENTATION_PORTRAIT)
{
    // Je viens du mode paysage et je bascule en portrait
    if( (fragCapteurName != null)) {
        ft = fr.beginTransaction();
        if (fragCapteurName.equals("DetailCapteurFragment")) ← On restaure le fragment
                                                                correspondant au nom de
                                                                classe sauvegardé
            ft.replace(R.id.ListeCapteursFragment, new ListeCapteursFragment());
        else if (fragCapteurName.equals("AccelerometerFragment"))
            ft.replace(R.id.ListeCapteursFragment, new AccelerometerFragment());
        ...
        else if (fragCapteurName.equals("TemperatureFragment"))
            ft.replace(R.id.ListeCapteursFragment, new TemperatureFragment());
        ft.commit();
    }
}
else {
    // Je viens du mode portrait et je bascule en mode paysage
    ListeCapteursFragment fgauche = new ListeCapteursFragment(); ← On restaure le fragment de liste des
                                                                    capteurs
    ft = fr.beginTransaction();
    ft.replace(R.id.ListeCapteursFragment, fgauche); ft.commit();
    if( (fragListeName != null)) {
        ft = fr.beginTransaction();
        if (fragListeName.equals("ListeCapteursFragment")) ←
                                                                On restaure le fragment de droite par le fragment
                                                                correspondant au nom de classe sauvegardé
            ft.replace(R.id.detailCapteurFragment, new InviteFragment());
        ...
        ft.commit();}
    }
}
} // FIN onCreate()

```



La classe MainActivity (suite)...

Méthode de sauvegarde de l'état (en cas de basculement de la vue)

@Override

```
protected void onSaveInstanceState(Bundle outState) {  
    Fragment fragListe =  
        getSupportFragmentManager().findFragmentById(R.id.listeCapteursFragment);  
    Fragment fragCapteur =  
        getSupportFragmentManager().findFragmentById(R.id.detailCapteurFragment);  
    if (fragListe != null)  
        outState.putString(this.FRAGLISTEPERSISTANT_KEY,  
                           fragListe.getClass().getSimpleName());  
    else  
        outState.putString(this.FRAGLISTEPERSISTANT_KEY, null);  
    if (fragCapteur != null)  
        outState.putString(this.FRAGCAPTEURPERSISTANT_KEY,  
                           fragCapteur.getClass().getSimpleName());  
    else  
        outState.putString(this.FRAGCAPTEURPERSISTANT_KEY, null);  
    super.onSaveInstanceState(outState);  
}
```

Sauvegarde des
noms de classe des
fragments chargés
dans l'activité



La classe MainActivity (suite)...

Méthode de remplacement d'un fragment, paramètres :
fragmentID = l'ID du fragment défini dans son layout,
fragment = l'objet Fragment de remplacement.

```
private void showFragmentByID(final int fragmentID, final Fragment
fragment)
{
    if (fragment == null)
        return;
    final FragmentManager fm = getSupportFragmentManager();
    final FragmentTransaction ft = fm.beginTransaction();
    ft.setCustomAnimations(android.R.anim.slide_in_left,
                           android.R.anim.slide_out_right);
    ft.replace(fragmentID, fragment);
    ft.addToBackStack(null);
    ft.commit();
}
```

Effet de glissement du
fragment de gauche à droite



La classe MainActivity (suite)...

Méthode surchargée de l'écouteur implémenté *ListeCapteursFragment.OnCapteurSelectedListener*

```
public void onCapteurSelecteur(int sensorType) {  
    if(getResources().getConfiguration().orientation == Configuration.ORIENTATION_LANDSCAPE){  
        Fragment fragment=  
            getSupportFragmentManager().findFragmentById(R.id.detailCapteurFragment);  
        switch(sensorType){  
            case Sensor.TYPE_ACCELEROMETER:  
                if (fragment != null)  
                    showFragmentById(R.id.detailCapteurFragment, (Fragment) new AccelerometerFragment());  
                break;  
                ... // etc pour tous les types de capteurs  
        }  
    }  
    else {  
        Fragment fragment =  
            getSupportFragmentManager().findFragmentById(R.id.ListeCapteursFragment);  
        switch(sensorType){  
            case Sensor.TYPE_ACCELEROMETER:  
                if (fragment != null)  
                    showFragmentById(R.id.ListeCapteursFragment, (Fragment) new AccelerometerFragment());  
                break;  
                ... // etc pour tous les types de capteurs  
        }  
    }  
}
```

Fragment de droite en mode paysage

Fragment unique en mode portrait



ANDROID

Le Manifeste avec son unique activité

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="net.michel.marie.projetcapteurliste"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="14"
        android:targetSdkVersion="19" />
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="net.michel.marie.projetcapteurliste.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Pause café ???

