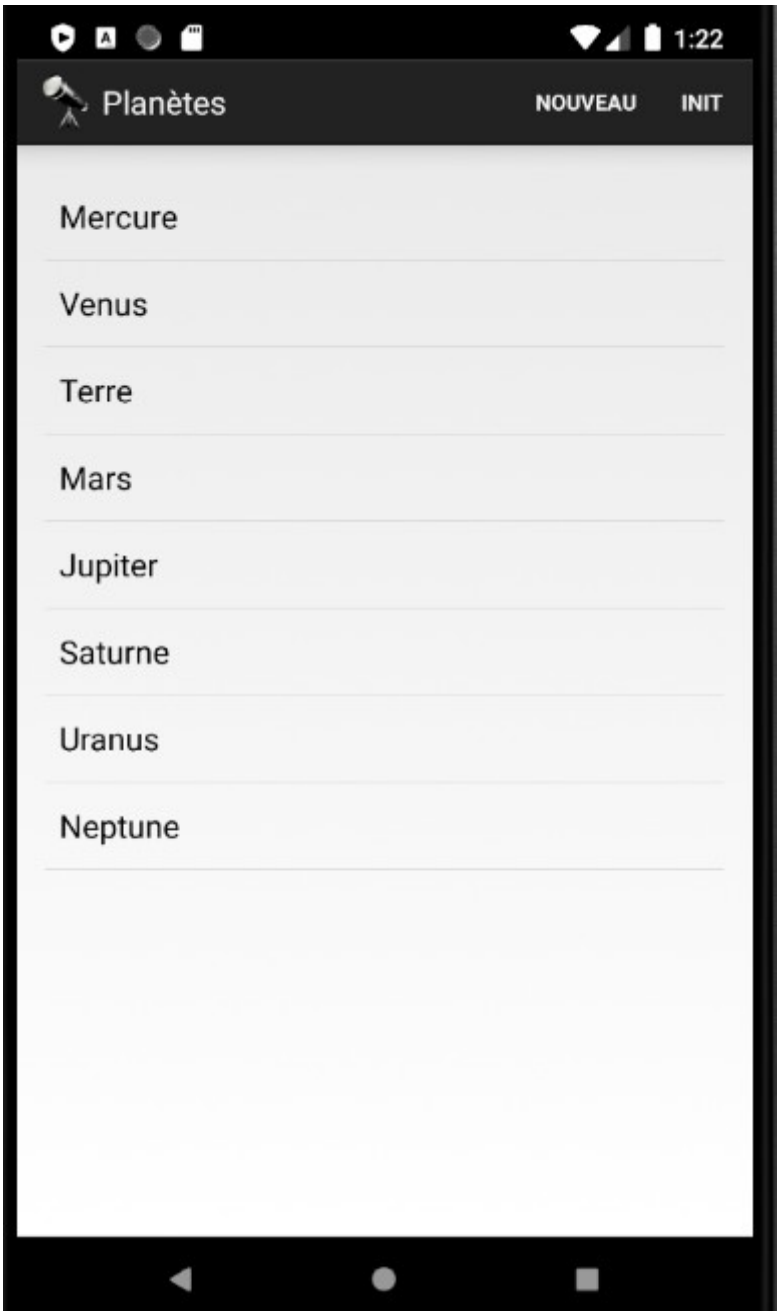
TD5 - Les listes d'items

Sommaire :

Introduction :

Le TP est en deux parties : comprendre un projet existant puis le modifier.

Voici le projet tel qu'il nous est donné :
On retrouve un bouton NOUVEAU qui ajoute de manière aléatoire un objet ainsi que un bouton INIT qui réinitialise l'application tel qu'elle l'est à son lancement.

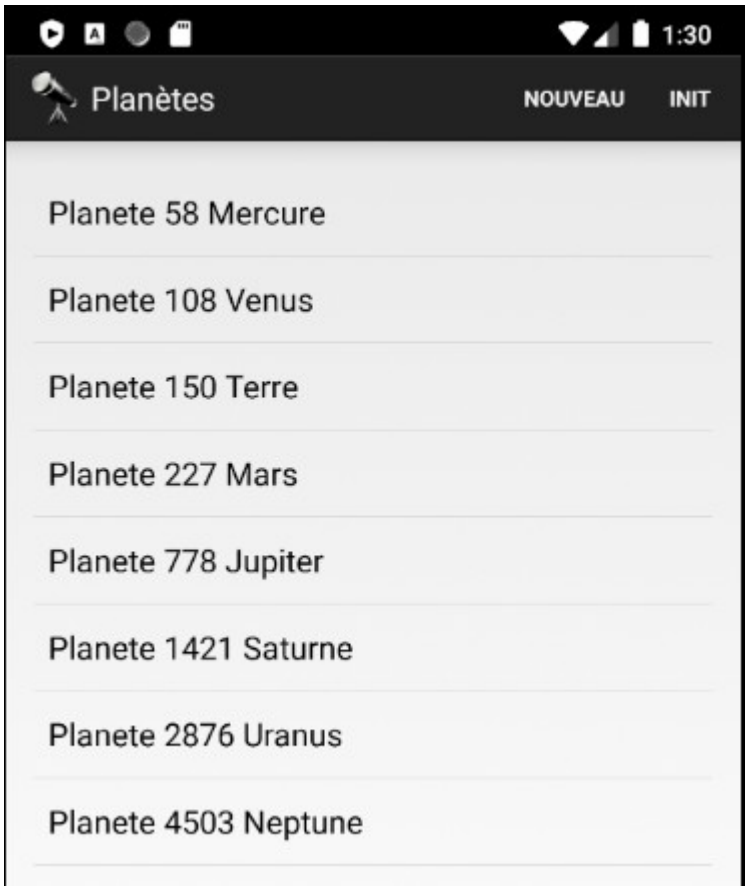


1.2.1. Fonctionnement de l'adaptateur standard

• Modifiez la méthode Planete.toString(), faites-lui retourner "Planete "+nom et constatez le résultat.

• Faites-lui retourner la distance au lieu du nom pour voir.

```
public String toString()
{
    return "Planete "+distance+" "+nom;
}
```



J'ai préféré afficher le nom **et** la distance afin de faciliter la compréhension du TP

1.2.2.Changer d'adaptateur

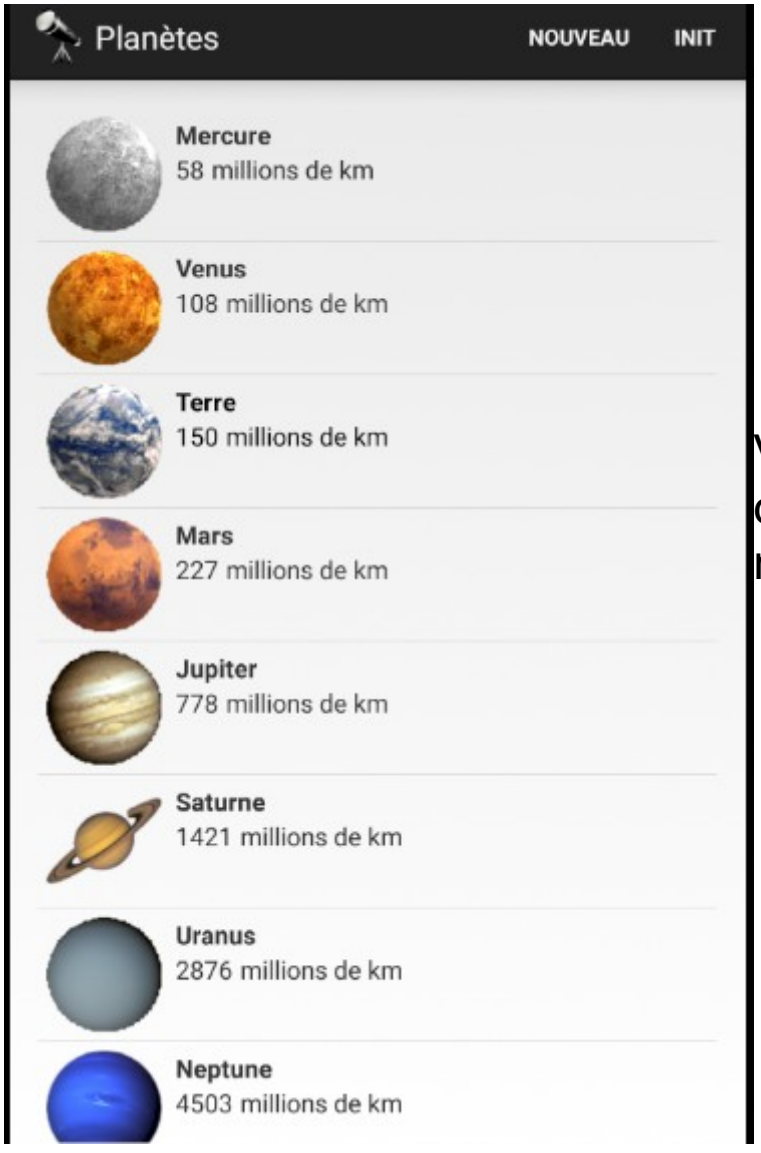
• Dans la méthode onCreate de l'activité, mettez les lignes qui créent l'adaptateur standard en commentaire et rajoutez seulement celle-ci :

```
/*adapter = new ArrayAdapter<>(this,
        android.R.layout.simple_list_item_1,
        android.R.id.text1,
        liste);*/
```

On met l'adaptateur en commentaire

```
adapter =new PlaneteAdapter( context: this, liste);
```

On ajoute le nouvel adaptateur



Voici le nouveau rendu de l'application avec ce nouvelle adaptateur.

1.2.3 . Édition ou création d'un item

• Relisez le cours de la semaine 3 : Transport d'informations dans un Intent, Extraction d'informations d'un Intent ainsi que Lancement avec attente de résultat.

• Dans la classeMainActivity, méthode onItemClick : programmez le lancement de l'activité EditActivity en rajoutant à l'intent un extra appelé identifiant et valant position 1 . Il indique la planète sélectionnée.
Lancez EditActivity par startActivityForResult.

```
@Override
public void onItemClick(AdapterView<?> parent, View v, int position, long id)
{
    // exemple 1 : afficher un message
    Toast.makeText( context: this,  text: "onItemClick: position=" + position, Toast.LENGTH_SHORT).show();
    Intent intent = new Intent( packageContext: this, EditActivity.class);
    intent.putExtra( name: "position",position);
    startActivityForResult(intent, RESULT_CANCELED);
```

• Dans la classe EditActivity, méthode onCreate, extrayez l'identifiant de l'intent, avec une valeur par défaut de -1. Vérifiez que ça marche : quand vous cliquez sur une planète, ça affiche bien l'EditActivity avec les informations de la planète. La méthode qui affiche les informations de la planète s'appelle setItem ; c'est la jumelle de celle de PlaneteView.

```
protected void onCreate(Bundle savedInstanceState) {
    // mettre en place l'interface
    super.onCreate(savedInstanceState);
    setContentView(R.layout.edit_activity);
    findViews();

    // FIXME aller chercher l'identifiant dans l'intent qui a lancé cette activi
    Intent intent = getIntent();
    identifiant = intent.getIntExtra( name: "position", defaultValue: -1);

    // changer le titre de la fenêtre selon qu'on édite ou qu'on crée un item
    if (identifiant < 0) {
        setTitle(R.string.nouveau);
    } else {
        setTitle(R.string.edit);
    }

    // afficher les informations éditables (nom,distance) de la planète
    setItem();
}
```

• Dans la classe MainActivity, méthode onMenuNouveau : effectuez le lancement de EditActivity, cette fois sans extra, ou alors avec -1.
• Dans la classe EditActivity, méthode onValider : elle est très incomplète. Cette méthode doit enregistrer les valeurs que l'utilisateur a tapé dans les vues. Alors il y a deux cas : identifiant correct ou négatif. Dans le premier cas, il faut aller chercher l'item dans la liste, dans le second cas, il faut créer un item et le rajouter à la liste.

```
public void onValider() {
    PlanetesApplication app = (PlanetesApplication) getApplicationContext();
    ArrayList<Planete> liste = app.getListe();

    // item concerné par l'activité d'édition
    Planete planete;

    // FIXME si l'identifiant est -1, alors il faut créer l'item, sinon il faut le prendr

    planete = new Planete();

    // récupérer les valeurs présentes dans les vues
    planete.setNom(etNom.getText().toString());
    planete.setDistance(Integer.parseInt(etDistance.getText().toString()));
    planete.setHabitabilite(rb.getRating());
    // on ne peut pas màj l'image, il faut faire autrement mais ce n'est pas demandé

    if(identifiant<0){
        liste.add(planete);
    } else {
        liste.set(identifiant,planete);
    }

    // TODO trier la liste sur la distance des planètes ok
    app.sortListe();
}
```

1.2.4. Ajout de la distance

• Dans le layout edit_activity.xml, rajoutez une vue pour saisir la distance. On ne doit pouvoir entrer que des entiers (définir android:inputType correctement).

```
<EditText
    android:id="@+id/item_planete_distance"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:ems="10"
    android:hint="Distance en millions de KM"
    android:inputType="number">

    <requestFocus />
```

• Dans la classe EditActivity, complétez findViews, setItem et onValider pour gérer la distance. Pensez à convertir les nombres en chaînes, et inversement, en particulier les entiers, sinon vous allez découvrir un nouveau cas de plantage de l'application.

```
protected void onCreate(Bundle savedInstanceState) {
    // mettre en place l'interface
    super.onCreate(savedInstanceState);
    setContentView(R.layout.edit_activity);
    findViews();
```

```
private void findViews() {
    ivImage = (ImageView) findViewById(R.id.item_planete_image);
    etDistance = (EditText) findViewById(R.id.item_planete_distance);
    etNom = (EditText) findViewById(R.id.item_planete_nom);
    rb = findViewById(R.id.ratingBar);
}
```

1.2.5. Modification de l'adaptateur

Le but est de changer la couleur de fond une ligne sur deux. Dans la méthode getView de PlaneteAdapter faites en sorte de changer la couleur de fond (background color en anglais).

L'entier qu'on fournit se construit à l'aide de la classe Color, par exemple Color.LTGRAY.

```
package fr.planetes;

import ...


public class PlaneteAdapter extends ArrayAdapter<Planete>
{
    public PlaneteAdapter(Context context, List<Planete> planetes) { super(context,  resource: 0, planetes); }

    @Override
    public View getView(int position, View convertView, ViewGroup parent)
    {
        // créer ou récupérer un PlaneteView

        PlaneteView planeteView = (PlaneteView) convertView;
        if (planeteView == null) {
            planeteView = PlaneteView.create(parent);
        }
        // TODO modifier la couleur de fond de l'item selon la parité de position
        if(position%2==0)
            planeteView.setBackgroundColor(Color.LTGRAY);

        // affecter les vues avec les valeurs de l'item n°position
        planeteView.setItem(super.getItem(position));
        return planeteView;

    }
```

Voici l'application avec l'ajout des couleurs si il s'agit d'un nombre pair.

Conclusion :

J’ai trouvé ce TD innovant, par rapport aux autres TD précédents puisqu’il s’agissait de reprendre un projet et de le comprendre de manière à le compléter.

Je n’ai pas pu faire la barre de notation (RatingBar) il m’était impossible de récupérer la valeur de la barre de notation afin de la stocker (Problème de Get & Set), l’application plante lorsque l’on clique sur les planètes pour leur mettre une note.