



Pragmatec
INGENIERIE INFORMATIQUE

Exercices Applications Android Avec Android Studio

Table des matières

<i>Section 1 Exercices – Mode d’emploi.....</i>	<i>3</i>
<i>Section 2 Installation de l’environnement de développement.....</i>	<i>4</i>
<i>Section 3 Création du projet.....</i>	<i>7</i>
<i>Section 4 Configurer pour utiliser le téléphone Android.....</i>	<i>10</i>
<i>Section 5 Se familiariser avec le mode Design.....</i>	<i>12</i>
<i>Section 6 Ajouter une vue – une Activity.....</i>	<i>15</i>
<i>Section 7 Gérer les styles et thèmes.....</i>	<i>18</i>
<i>Section 8 Gérer les images.....</i>	<i>20</i>
<i>Section 9 Gérer les icônes.....</i>	<i>22</i>
<i>Section 10 Une application Quiz.....</i>	<i>23</i>
<i>Section 11 Créer un compte Google Play.....</i>	<i>37</i>

Section 1 Exercices – Mode d’emploi

Pour la prise en main de Android Studio et des technologies Android nous allons créer une solution nommée « lelab »

Cette application va être faite en suivant les étapes suivantes :

- Création du projet
- Réaliser le compteur
- Navigation
- Design et style
- Tabs et listes

Section 2 Installation de l'environnement de développement

Le lien suivant peut être utilisé pour tout problème :

<https://openclassrooms.com/courses/creez-des-applications-pour-android/installation-et-configuration-des-outils>

2.1 Installation du JDK Java Development Kit

Cette bibliothèque contient la partie runtime, la JRE avec la machine virtuelle. Contient en plus des outils propres au développement, compilateurs ...


Aller sur le lien

<https://www.oracle.com/java/technologies/javase-downloads.html>

Cliquer sur ce lien :



Choisir ensuite

Windows x64 Installer	162.11 MB	 jdk-14.0.2_windows-x64_bin.exe
-----------------------	-----------	--

Installer le JDK

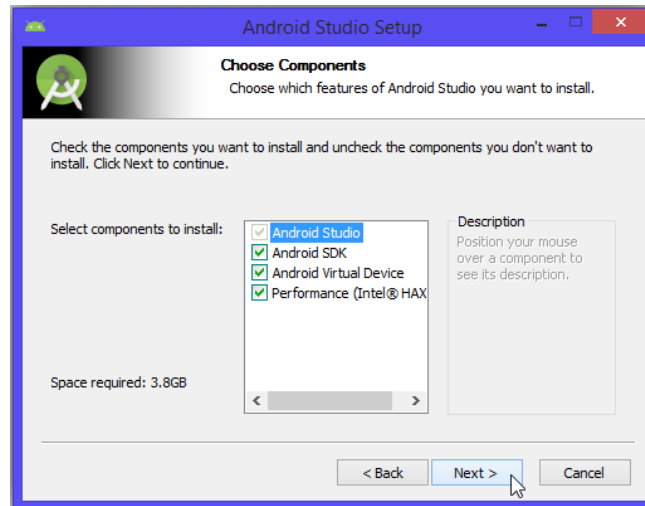
2.2 Installation de Android Studio

Aller sur le site

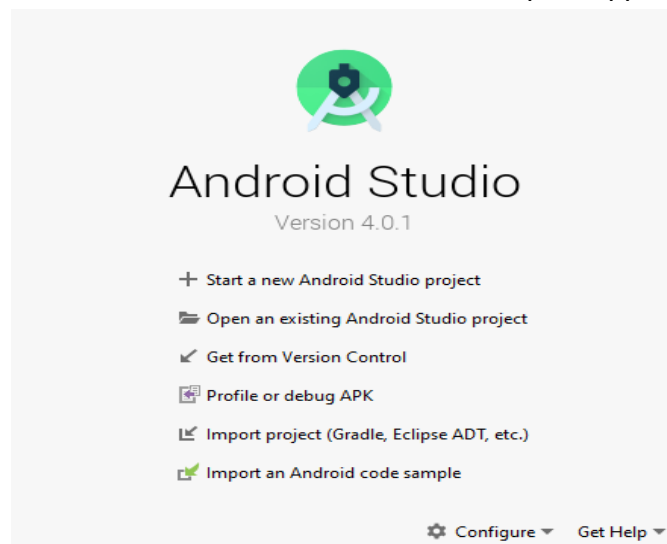
<https://developer.android.com/studio>

A green rectangular button with the text "DOWNLOAD ANDROID STUDIO" in white, bold, uppercase, sans-serif font.

Pendant l'installation, accepter les choix par défaut. Dans la fenêtre suivante accepter ces choix :

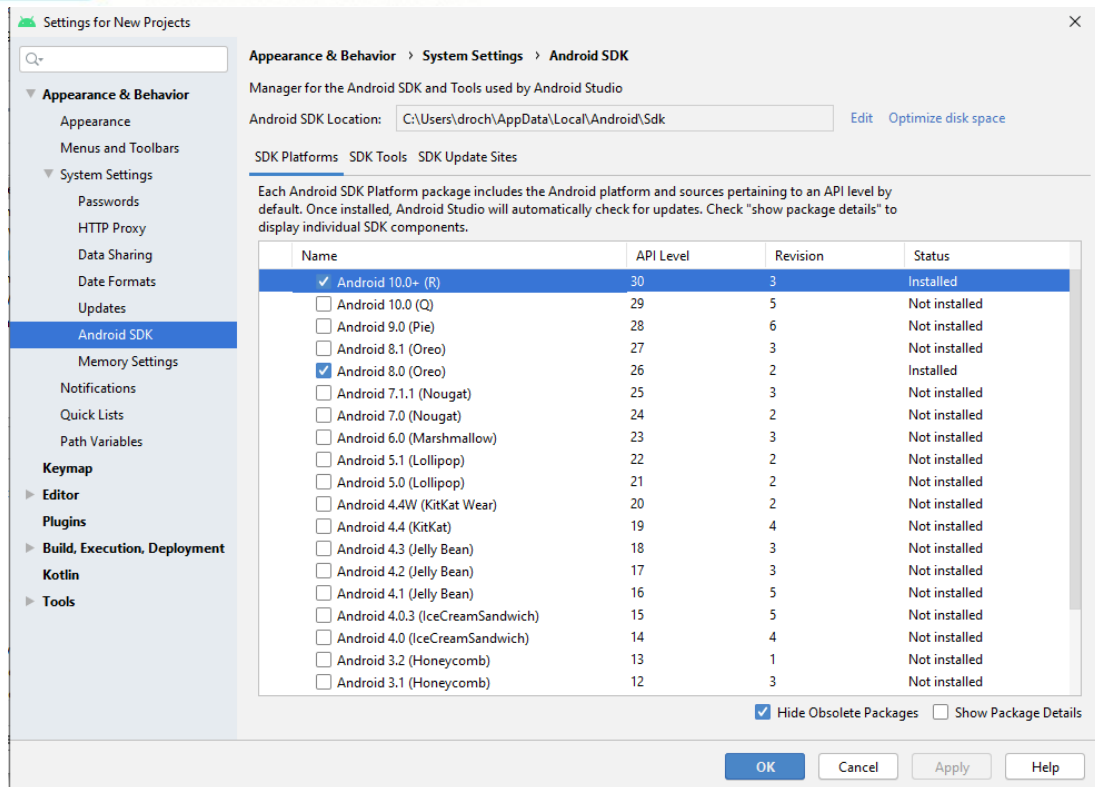


Au lancement de Android studio la fenêtre suivante peut apparaître :



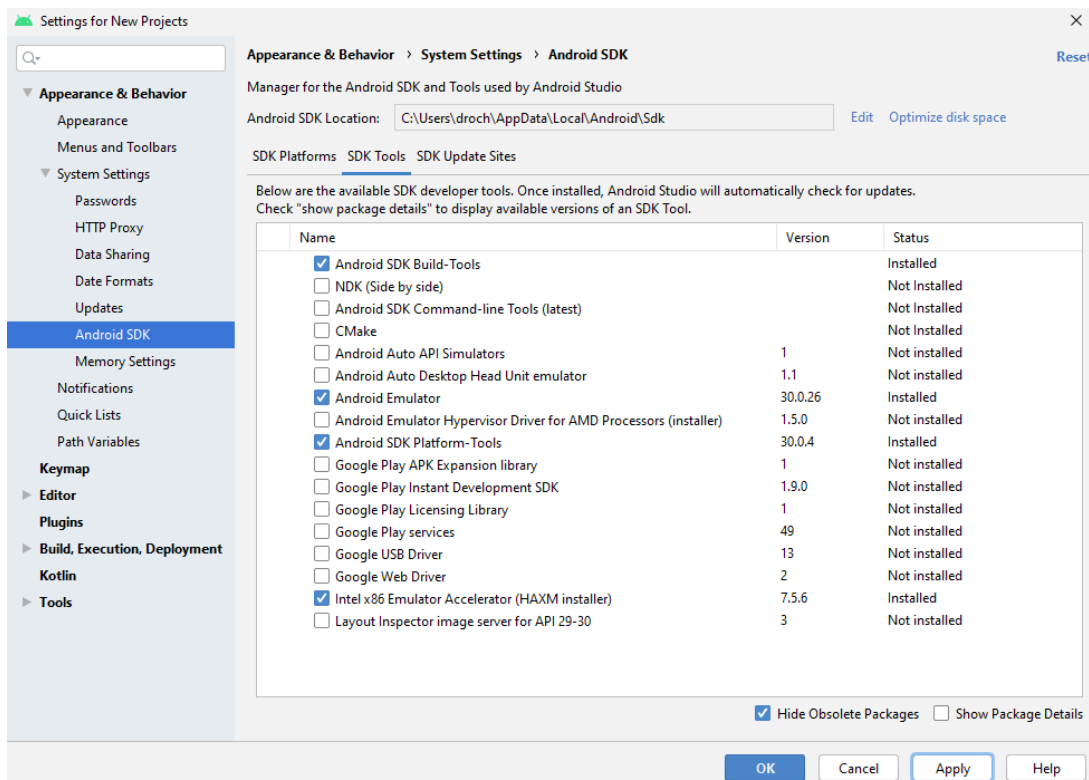
Dans ce cas choisir Configure puis SDK Manager.

Sinon, si l'IDE complète est déjà affichée, choisir Tools/SDK Manager



Cocher au minimum Android 8,0 puis Android 10 comme ci-dessus puis bouton Apply.

Puis cliquer sur l'onglet SDK Tools :



Faire les choix comme ci-dessus puis Apply

Section 3 Création du projet

3.1 But

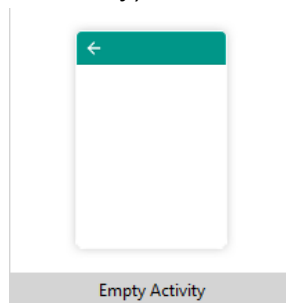
- Créer le projet sous Android Studio,
- configurer un simulateur

3.2 Exercice

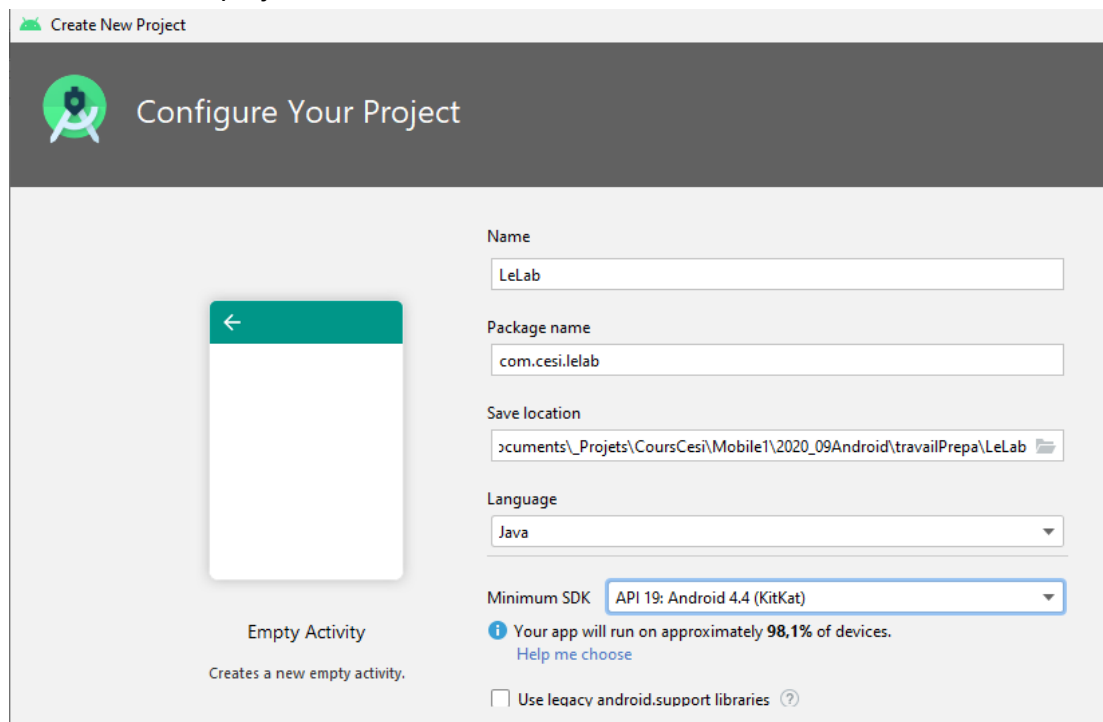
Sous Android Studio , créer un projet par
Fichier > Nouveau >projet

+ Start a new Android Studio project

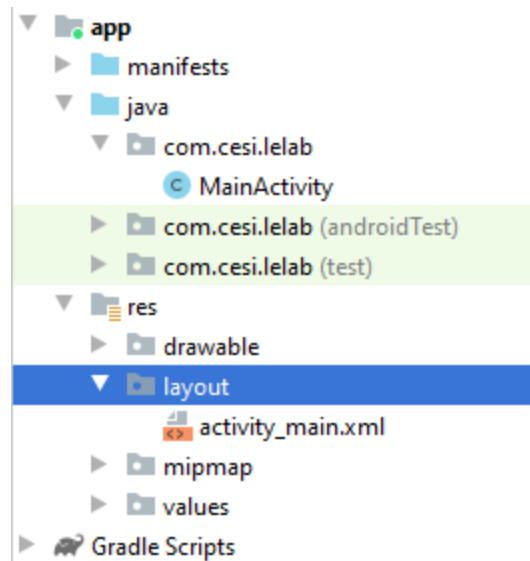
ou par Choisir un projet (une Activity) vide



Nommer le projet « LeLab »



On obtient ce qui suit :



Regarder le contenu de MainActivity.java. C'est le point d'entrée de l'application

Puis activity_main.xml. Contient la description de la vue mentionnée dans MainActivity.java


Le fichier AndroidManifest.xml décrit les pages (Activity) constituant l'application. Contient aussi l'utilisation des styles et des thèmes.


Dans activity_main.xml ajouter un bouton.

Nous allons maintenant simuler cette page.

Ouvrir la fenêtre d'ajout d'un device virtuel : Tools/AVD Manager

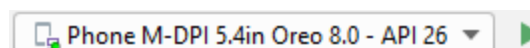
Dans la fenêtre choisir ce matériel :

Type	Name	Play Store	Resolution	API
	Phone M-DPI 5.4in Ore...		480 × 854: mdpi	26

Clic sur  Download ▼ s'il apparaît

Lancer le device depuis cette fenêtre ou depuis le haut de l'IDE Android Studio.

Puis clic sur



Il se peut que la configuration Windows du PC empêche le bon fonctionnement du mode HAXM. Une popup d'erreur est alors affichée.

Tenter alors la manipulation suivante :

Dans Windows dans le menu Démarrer, ouvrez le Panneau de configuration. Ensuite, cliquez sur le Menu Programmes. Cliquer sur le texte bleu [Programmes et fonctionnalités](#), cliquez sur le lien Activer ou désactiver des fonctionnalités Windows. Dans la fenêtre affichée cocher Hyper V et Plateforme hyperviseur.

Redémarrer le PC.

Si les textes de l'application sont petits il faut aller dans la configuration du téléphone simulé et agrandir l'affichage des textes.

Section 4 Configurer pour utiliser le téléphone Android

4.1 But

- Permettre de charger l'application sur une vraie cible
- Tester sur la cible

4.2 Exercice

NE PAS raccorder pour l'instant le téléphone au PC.

Sur la station de développement (le PC) il faut installer le driver compatible avec le matériel Android.

Le lien suivant fournit les drivers en fonction du matériel.

<https://developer.android.com/studio/run/oem-usb#Drivers>

Pour le matériel Motorola MOTO, le driver est joint à la formation.

Il est disponible à l'adresse <https://support.motorola.com/us/en/solution/MS88481>

L'installer.

Avant de raccorder le téléphone au PC en USB, il faut configurer le téléphone :

Dépend d'un tel à l'autre, en fonction de la version Android installée.

Sur MOTO,

- cliquer sur Paramètres
- Aller sur Système
- Aller sur A propos du téléphone
- Aller sur Numéro de build
- Appuyer 7 fois sur cet item Numéro de build. Si au premier appui, le tel indique « Inutile vous êtes déjà un développeur », inutile de continuer.

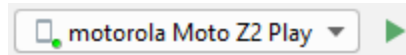
Cette manipulation bizarre indique à Android que l'on souhaite utiliser la liaison USB pour transférer des applications.

Suite à cette manipulation, retourner en arrière, page Système. L'item « Options développeurs » est maintenant visible.

Aller dans cette vue et activer « Débogage USB »

Relier le tel en USB au PC.

Dans Android Studio, le nom du matériel simulé laisse place au téléphone au bout de quelques secondes.



Lancer le chargement, flèche verte.

L'application doit être visible sur le téléphone ...

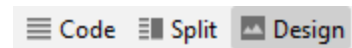
Section 5 Se familiariser avec le mode Design

5.1 But

- Android Studio propose un mode dessin pour définir les pages, quelques informations ici pour l'utiliser.
- Le site suivant fournit une aide :
<https://developer.android.com/studio/write/layout-editor#intro>

5.2 Exercice

- Afficher l'activité activity_main.xml
- Employer les boutons suivants
- Ouvrir le menu et choisir Blueprint

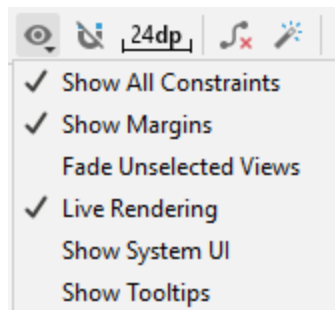


Blueprint montre des informations cachées en mode Design

- Régler la Default Margin à 24 dp . C'est l'espace par défaut laissée entre plusieurs objets graphiques.



- Cliquer sur l'oeil et régler ainsi :



Pour se familiariser nous utilisons l'exemple
<https://developer.android.com/training/basics/firstapp/building-ui>

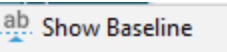
- Supprimer les objets existants, ne garder que le ConstraintLayout
- Ajouter un objet Text/Plain Text dans l'Activity



- Mettre les contraintes de positionnement suivantes :



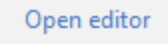
Ajouter un bouton à droite du texte. Placer une contrainte horizontale entre le bouton et le texte

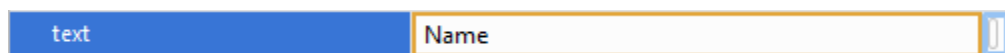
- Pour contraindre le bouton à être au centre de la zone de texte : clic sur le bouton, menu contextuel et choix de  Puis relier les parties centrales du bouton et du texte pour obtenir ceci :



Tester le résultat.

Les textes affichés doivent être dans un fichier de ressource, voici comment procéder :

- ouvrir le fichier res/values/strings.xml
- 
- clic sur + pour ajouter une chaîne
- créer la clé editMessage et valeur « Entrer un message »
- créer la clé btnSend et valeur « Envoyer »
- Retourner à la vue activity_main.xml
-



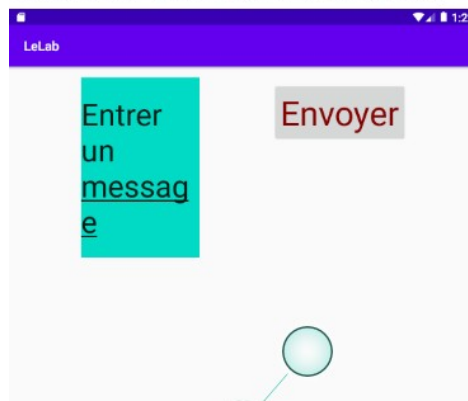
- désigner le texte. Dans la partie Properties, chercher et clic sur la partie droite. La fenêtre de ressource s'ouvre, choisir editMessage. Regarder le résultat. Faire les mêmes actions pour le bouton

Tester le résultat.

Il peut ressembler à ceci :

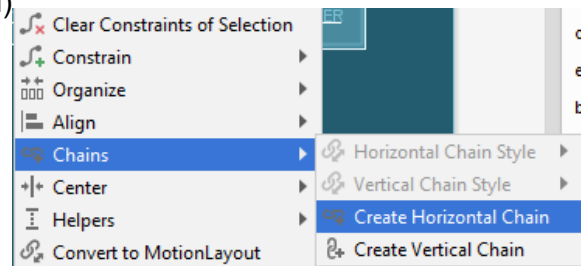


Android Emulator - phone_mdpi_5_4in_oreo_8_0_-_api_26:5554



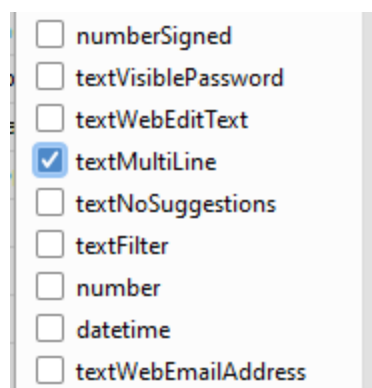
Nous améliorons l'affichage de l'objet saisie de texte en faisant en sorte qu'il occupe au maximum la largeur du support.

- Sélectionner les 2 objets (touche shift pour le second), puis menu contextuel et choix suivant (en bleu)



- Sur le editText, mettre layout_width à 0dp

Si l'on souhaite réaliser une saisie de texte sur plusieurs lignes, cliquer sur le drapeau de la propriété inputType du EditText



Section 6 .Ajouter une vue – une Activity

6.1 But

- Créer une vue sur le projet
- le site <https://developer.android.com/studio/write/layout-editor#intro> fournit une aide à l'emploi de Android Studio
- passage d'une vue à une autre

6.2 Exercice

Ajouter une vue correspond à « ajouter une Activity » au sens de Android.

Nous allons ajouter un événement au bouton Envoyer.

- Dans `mainActivity.java` ajouter le code

```
/** Appelé quand l'utilisateur clique sur Envoyer */
public void sendMessage(View view) {
    // Code appelé dans ce cas
}
```

- Dans la vue `activity_main.xml`, sélectionner le bouton et affecter cette propriété :

onClick	sendMessage
orientation	MainActivity
outlineProvider	sendMessage

- Changer l'id du text en editText

id	editText
----	----------

Il faut coder l'événement.

Nous utilisons une constante . Le code complet est celui-ci :

```
public class MainActivity extends AppCompatActivity {  
    public static final String EXTRA_MESSAGE = "com.cesi.lelab.MESSAGE";  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
    }  
    /** Appelé quand l'utilisateur clique sur Envoyer */  
    public void sendMessage(View view) {  
        Intent intent = new Intent(this, DisplayMessageActivity.class);  
        EditText editText = (EditText) findViewById(R.id.e);  
        String message = editText.getText().toString();  
        intent.putExtra(EXTRA_MESSAGE, message);  
        startActivity(intent);  
    }  
}
```

Les objets de la classe Intent sont utilisés pour partager des informations entre plusieurs Activities. Ici utilisé pour fournir une information à la future Activity.

DisplayMessageActivity du code ci-dessus n'existe pas encore, nous allons la créer ...

Créer la nouvelle activity pour afficher le message saisi.

- Dans la fenêtre Projet, menu app/New/Activity/Empty Activity
- Donner le nom DisplayMessageActivity

3 choses ont été faites par Android Studio :

- création de DisplayMessageActivity.java
- création de res/layout/activity_display_message.xml
- ajout de la référence à cette Activity dans AndroidManifest.xml

Tester ce résultat

La seconde page (Activity) s'affiche, pour l'instant vide.

Pour y mettre un contenu, ajouter dans la partie xml de la page un TextView avec l'identifiant « textView »


Puis ajouter le code suivant dans DisplayMessageActivity.java

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_display_message);
    // Récupérer l'objet Intent qui a démarré l'Activity et extraire la
    chaîne identifiée par la constante EXTRA_MESSAGE
    Intent intent = getIntent();
    String message = intent.getStringExtra(MainActivity.EXTRA_MESSAGE);

    // Chercher le TextView de cette Activity
    TextView textView = findViewById(R.id.textview);
    textView.setText(message);
}
```

Tester ce résultat

La seconde page (Activity) s'affiche, avec le contenu du texte de la page précédente.

Pour revenir en arrière, l'icône  peut être utilisée. Pour mettre ce retour en arrière dans l'entête : dans le fichier Manifeste indiquer à DisplayMessageActivity qui est l'Activity parente par ce code :

```
<activity android:name=".DisplayMessageActivity"
    android:parentActivityName=".MainActivity"></activity>
```

Tester ce résultat.

Section 7 Gérer les styles et thèmes

7.1 But

- Harmoniser la présentation de l'application
- le site <https://developer.android.com/guide/topics/ui/look-and-feel/themes> fournit une aide

7.2 Exercice

Dans Android, le terme View désigne un élément graphique – widget dans d'autres langages. Ex Button, TextView ...

7.2.1 Un style Android

Un style est une collection d'attributs qui spécifie l'apparence de View. On peut ainsi spécifier la couleur, la taille ...

Un style est défini dans un xml de res/values, généralement dans styles.xml

Ajouter dans styles.xml

```
<style name="GreenText" parent="TextAppearance.AppCompat">  
  <item name="android:textColor">#008000</item>  
  <item name="android:background">#e000e0</item>  
</style>
```

Puis dans activity_main.xml affecter ce style aux Views.

style

Tester le résultat.

Un style peut hériter d'un autre. 2 façons possibles :

- en utilisant l'attribut parent

```
<style name="GreenLargeText" parent="@style/GreenText">  
  <item name="android:textSize">40dp</item>  
</style>
```

- en utilisant un point

```
<style name="GreenText.Large">  
  <item name="android:textSize">30dp</item>  
</style>
```

7.2.2 Un thème Android

Un style est donc appliqué individuellement à une View (un objet graphique).

Un thème est un style qui peut être appliqué à toute l'application, ou à certaines Activités choisies. « android:theme » est à placer aux bons endroits

Voici le code du Manifest, pour exemple

```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"

    android:theme="@style/GreenText">

    <activity android:name=".DisplayMessageActivity"
        android:parentActivityName=".MainActivity"

        android:theme="@style/GreenText.Large"></activity>

    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

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

Section 8 Gérer les images

8.1 But

- Utiliser des images : drawable resources
- le site <https://developer.android.com/guide/topics/resources/drawable-resource> fournit une aide

8.2 Exercice

Les formats supportés sont png, jpp, gif. Le format png est à préférer.

Les images peuvent être contenues dans des ImageView ou des ImageButton.

Elles peuvent également constituer le background de composants.

Mode opératoire :

- faire glisser un fichier .png depuis un navigateur de fichiers vers le nœud res/drawable de Android Studio
- Le fichier est identifié en v24. 24 signifie une définition de couleur sur 24 bits, soit 3 octets.
- Dans la vue activity_main placer un image button et un image view. Y placer le png.
- Tester

Le format WebP, créé par Google permet de réduire la taille mémoire des images.

Faisons un essai :

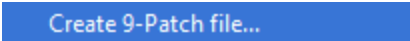
- sur le fichier png, menu contextuel et choix de Convert to WebP
- dans la fenêtre qui s'affiche choisir un taux de conversion acceptable.
- Tester la vue

Le format .9.png, Nine-Patch File permet de définir quelles sont les régions qu'il est possible de stretch (redimensionner) .

<https://developer.android.com/studio/write/draw9patch>

- Placer un fichier .png dans res/drawable

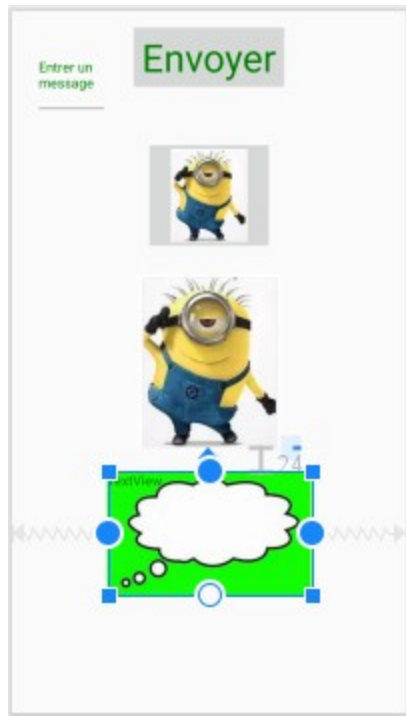
-



- menu contextuel sur ce fichier pour choisir cocher Show content



- agir sur les lignes de limitations. Il y a les lignes de limitation du stretch et de limitation de la zone de tracé de texte
- Dans activity-main.xml, ajouter un TextView
- A la propriété background, affecter l'image .9.png
-



Le résultat doit ressembler à ceci, le texte doit être correctement affiché dans la bulle :



Section 9 Gérer les icônes

9.1 But

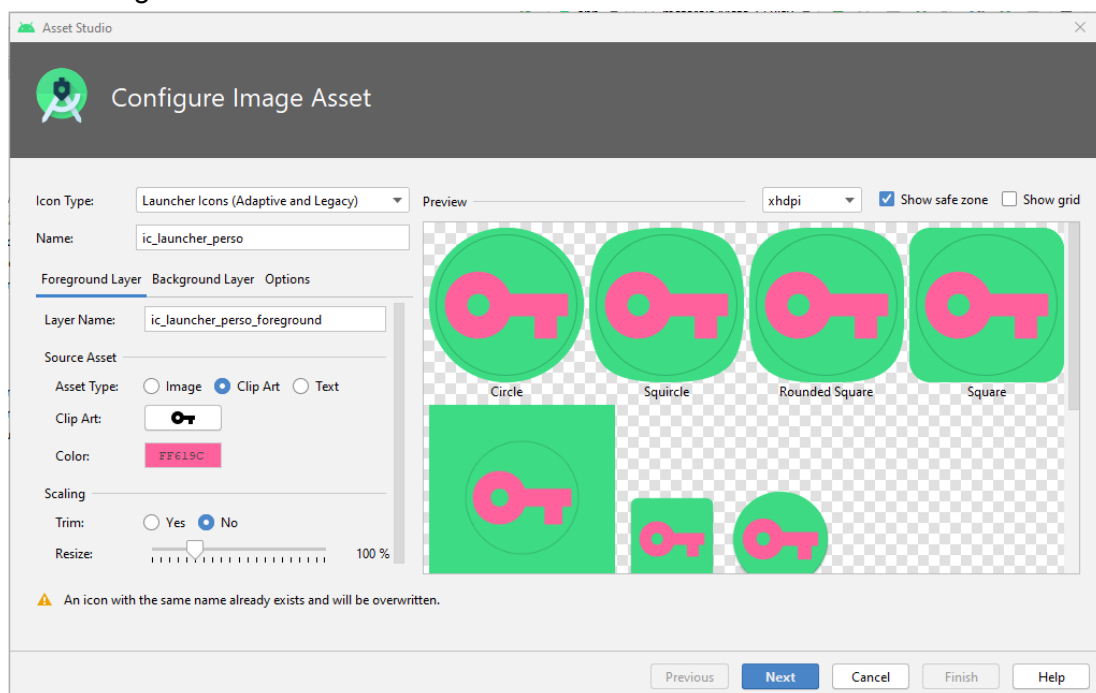
- Générer des icônes et les employer dans l'application
- <https://developer.android.com/studio/write/image-asset-studio>

9.2 Exercice

Nous allons créer des icônes pour représenter l'application dans la liste des applications.

Sur le nœud app, menu contextuel, choix New/Image asset

Configurer l'icône comme ceci :



Il faut éditer AndroidManifest.xml

Modifier comme suit :

```
android:icon="@mipmap/ic_launcher_perso"
android:roundIcon="@mipmap/ic_launcher_perso_round"
```

Section 10 Une application Quiz

10.1 But

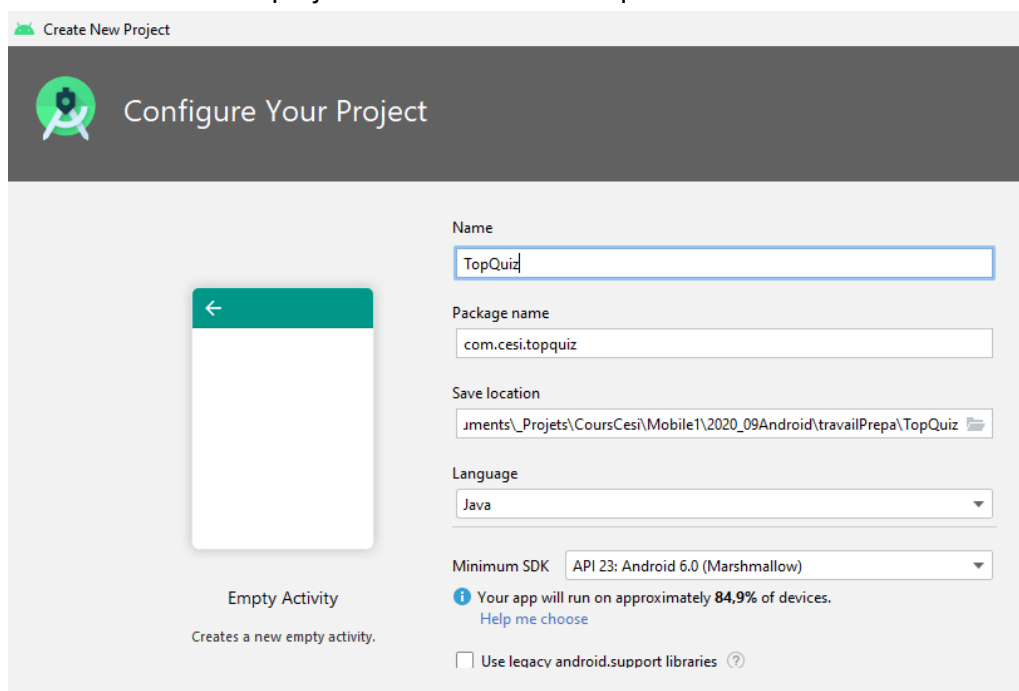
- Capitaliser ce que l'on vient de voir avec une autre application
- Provient du site <https://openclassrooms.com/fr/courses/4517166-developpez-votre-premiere-application-android>

10.2 Exercice

Nous allons créer une application sur plusieurs pages (Activities).

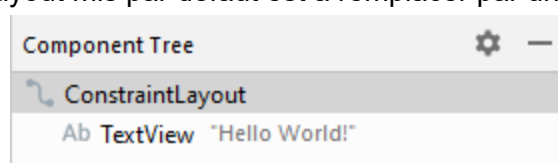
10.2.1 Initialiser le projet

Créer un nouveau projet avec les caractéristiques suivantes :



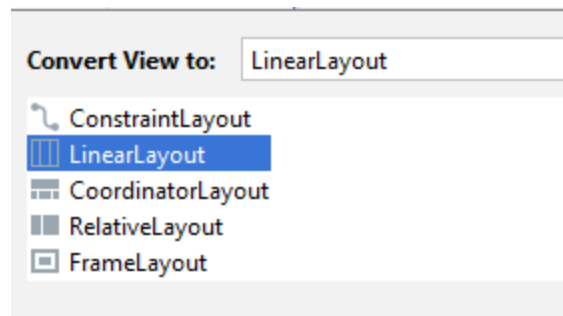
Ouvrir activity_main.xml.

Le ConstraintLayout mis par défaut est à remplacer par un LinearLayout Vertical.





Désigner Constraint Layout et appliquer un Convert View



Créer ceci :

Cette page est composée :

- du linear layout vertical
- d'un textView qui prend toute la largeur du parent. Le texte « Bienvenue, quel est votre prénom » provient d'une ressource string
- d'un Plain Text. qui prend toute la largeur du parent. Le texte « Entrer votre prénom » (prop hint) provient d'une ressource string
- d'un Bouton Centré sur la largeur (prop layout_gravity). Le texte « Au jeu ! » provient d'une ressource string
- adapter le padding des éléments pour les séparer les uns des autres

10.2.2 Ajout de listener

On veut éviter que le bouton soit cliqué avant que l'utilisateur ait saisi un prénom.

Dans MainActivity.java :

- déclarer 2 attributs

```
private Button mPlaybutton;  
private EditText mEditText;
```

- récupérer les widgets et rendre inactif le bouton

```
mPlaybutton = findViewById(R.id.button);  
mPlaybutton.setEnabled(false);  
mEditText = findViewById(R.id.editFirstName);
```

- ajouter un listener sur le EditText pour être averti quand il n'est pas vide.
Laisser faire Android Studio qui va pré remplir le code

```
mEditText.addTextChangedListener(new TextWatcher())
```

- coder la partie onTextChanged. Il s'agit de réaliser du code pour alimenter
`mPlaybutton.setEnabled()` avec le bon paramètre.
Pour aider il est possible de mettre des messages dans un log :
`Log.i(« main », « votre message »);`
- Peut on supprimer les méthodes `beforeTextChanged()` et `afterTextChanged()` ?

On souhaite réaliser une action sur clic du bouton.

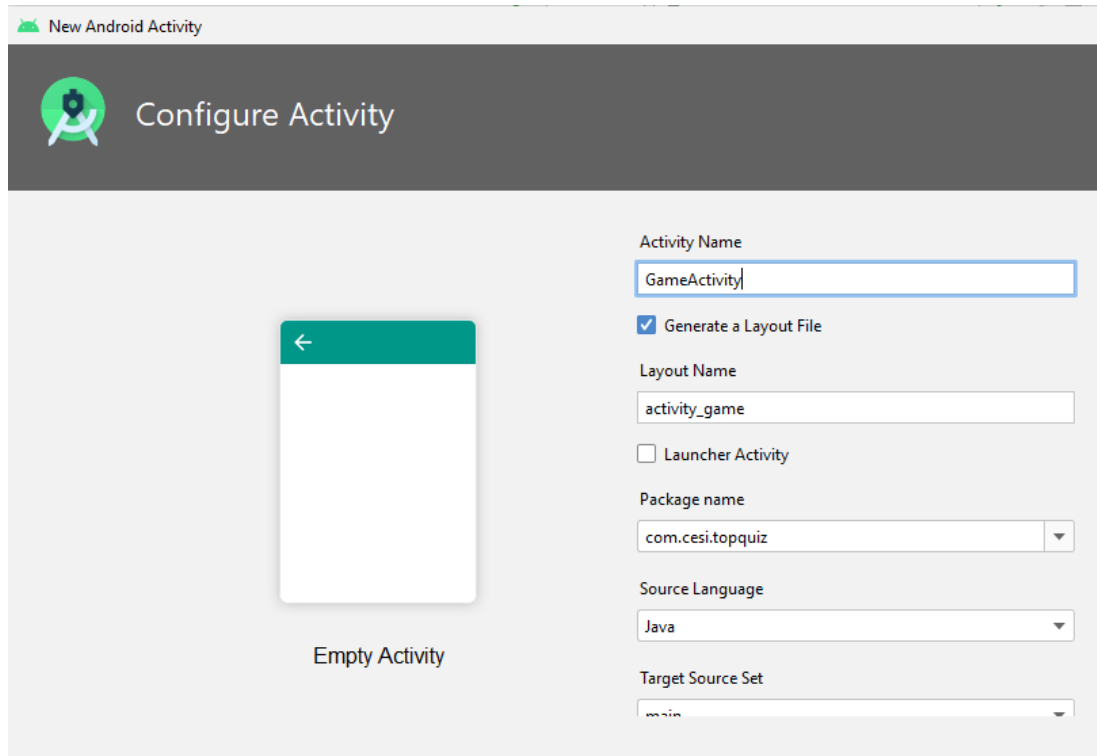
Dans MainActivity.java créer la méthode

```
public void onPlay(View view)
```

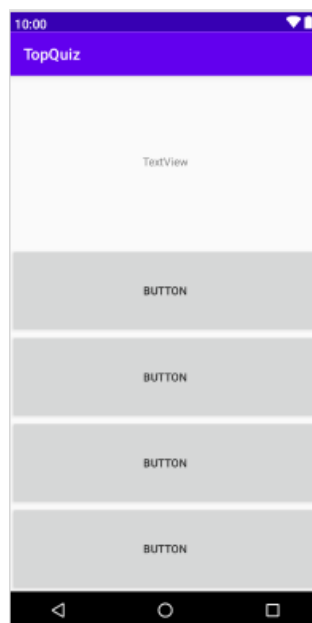
Dans activity_main.xml associer cette méthode à l'événement `OnClick`

10.2.3 Ajout de l'Activity Game

Depuis le nœud app, choisir new/Activity/Empty Activity et saisir le nom : GameActivity



Le but est d'obtenir ceci :



- Cette page contient : un LinearLayout vertical
- un TextView

- 4 boutons

Le TextView et bouton ont un `layout_height` à 0dp.

Ils ont un `layout_weight` permettant de se répartir sur la hauteur.

Le TextView à `layout_weight` à 2, les autres à 1.

La hauteur du TextView est donc de $2 / (2 + 1 + 1 + 1 + 1)$ soit 2/6

La hauteur de chaque bouton est 1 / 6

Pour que cette page soit appelée, il faut ajouter du code dans `OnClick()`.

Le code, déjà vu est le suivant :

```
Intent gameActivity = new Intent(MainActivity.this, GameActivity.class);
startActivity(gameActivity);
```

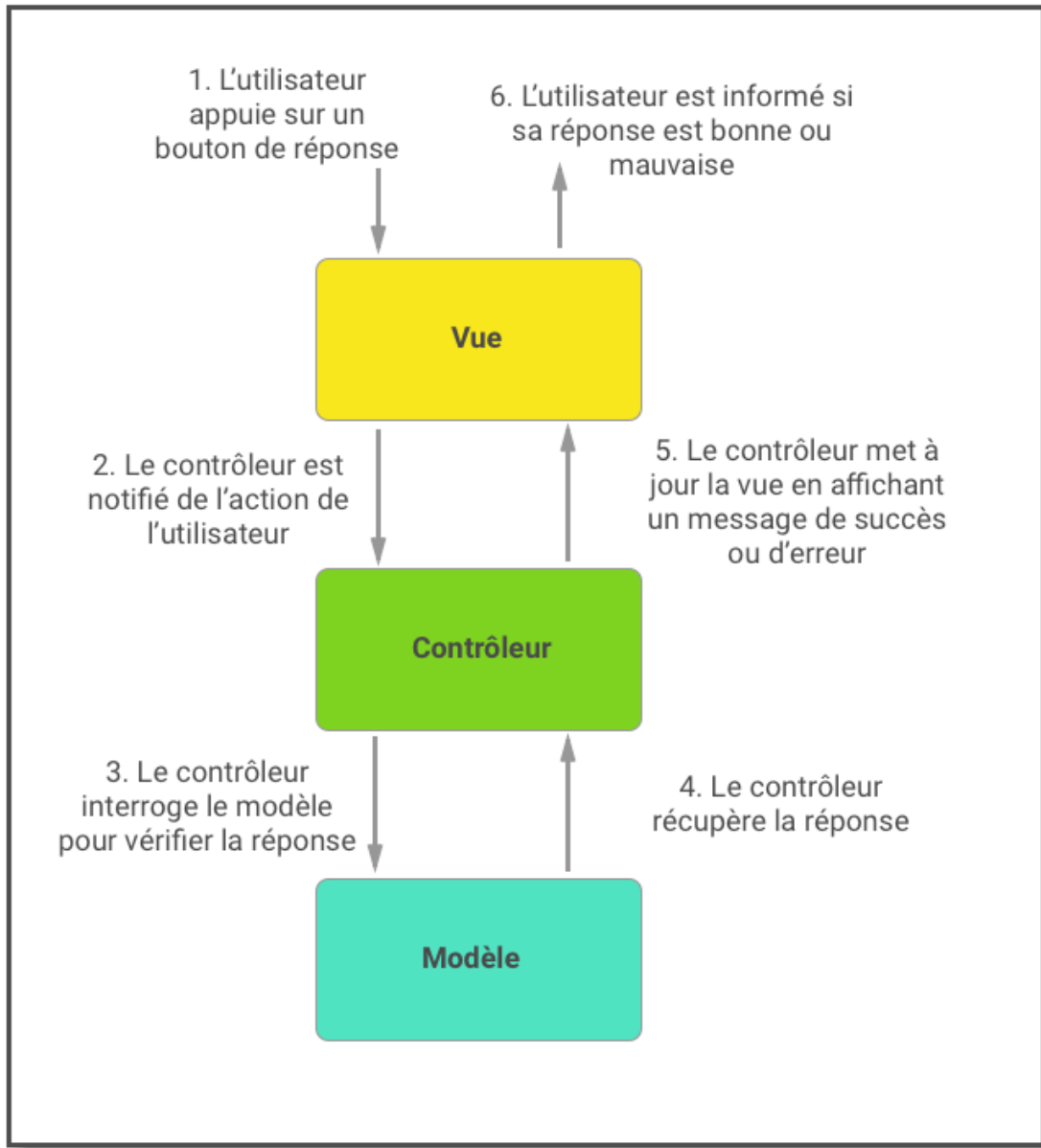
Tester cette page

Modifier `Androidmanifest.xml` pour disposer dans le titre de la page `GameActivity` de la flèche de retour.

10.2.4 Mise en place d'un modèle MVC

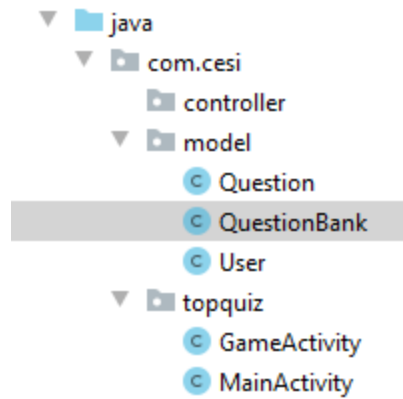
On souhaite développer proprement en suivant l'architecture MVC.

Pour ce jeu, cela consiste à ce qui suit (source openclassroom)

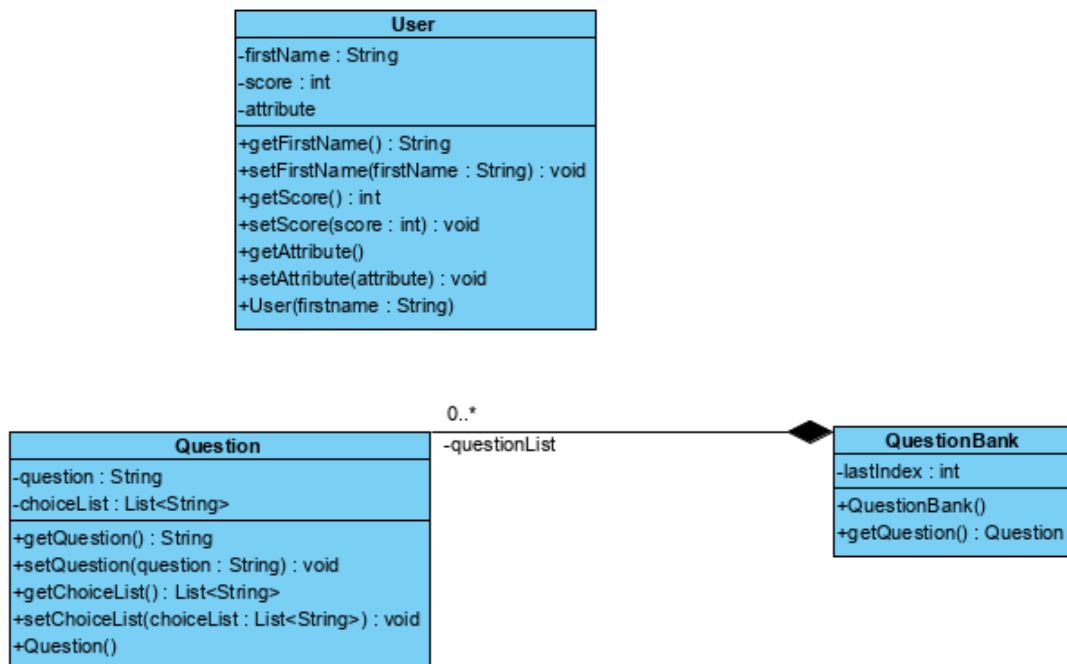


10.2.5 La couche Model

Créer un package nommé Model.



Créer dans ce package les classes correspondant au diag UML :



Dans GamaActivity.java créer une méthode generateQuestions() avec le code suivant :

```

Question question1 = new Question("Who is the creator of Android?",
    Arrays.asList("Andy Rubin",
        "Steve Wozniak",
        "Jake Wharton",
        "Paul Smith"),
    0);
  
```

```

Question question2 = new Question("When did the first man land on the moon?",
    Arrays.asList("1958",
        "1962",
        "1967",
        "1969"),
    3);

Question question3 = new Question("What is the house number of The Simpsons?",
    Arrays.asList("42",
        "101",
        "666",
        "742"),
    3);

return new QuestionBank(Arrays.asList(question1,
    question2,
    question3));

```

Dans le même fichier créer la propriété questionBank, et appeler generateQuestions() dans onCreate().

```
questionBank = generateQuestions();
```

10.2.6 La couche Controller

Elle est matérialisée par GameActivity.java

Créer maintenant dans ce fichier

```
private void displayQuestion()
```

Cette méthode utilise getQuestion() déjà codée et affecte un texte au textView et aux boutons.

Appeler displayQuestion() dans onCreate()

Tester.

Il faut maintenant associer aux 4 boutons la gestion d'un événement. On souhaite coder l'événement qu'une seule fois, pour les 4 boutons.

Créer la méthode **public void** onAnswer(View view)

et l'utiliser dans activity_game.xml

Il faut trouver un moyen dans le code de l'événement quel est le bouton qui a été cliqué : utiliser la propriété Tag.

Mettre un tag différent sur chaque bouton (ex 0 à 3).

Dans onAnswer(), view.getTag() permet de récupérer la valeur.

Tester la valeur reçue avec l'index de la bonne réponse à la question.

Pour afficher un message bulle, utiliser l'instruction

```
Toast.makeText(this, "Correct !", Toast.LENGTH_SHORT).show();
```

10.2.7 Affichage du score

Faire en sorte que suite à une réponse, correcte ou incorrecte, l'application pose la question suivante.

Quand il n'y en a plus, afficher le score.

Le code suivant ouvre une fenêtre popup

```
AlertDialog.Builder builder = new AlertDialog.Builder(this);

builder.setTitle("Parfait !")
    .setMessage("Votre score est " + score)
    .setPositiveButton("OK", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            finish();
        }
    })
    .create()
    .show();
```

10.2.8 Affichage du score dans la 1ère page

On va récupérer le score disponible dans la page 2 et le fournir à la page 1.

Le TextView du haut contiendra ce score.

C'est encore un objet Intent qui va permettre de véhiculer l'information.

L'appel de la page 2 ne se fait plus par `startActivity()`

mais par `startActivityForResult()`

Donc dans MainActivity.java, remplacer `startActivity()` par `startActivityForResult(gameActivity, ACTIVITY_CODE);`

Déclarer en tête de classe la constante

```
public final int ACTIVITY_CODE = 109;
```

Dans MainActivity.java, là où se trouve le code finish() on va ajouter le code pour placer le score dans un Intent :

```
Intent intent = new Intent();
intent.putExtra(BUNDLE_EXTRA_SCORE, score);
setResult(RESULT_OK, intent);
finish();
```

Déclarer en tête de classe la constante

```
public static final String BUNDLE_EXTRA_SCORE = "BUNDLE_EXTRA_SCORE";
```

Retour dans MainActivity.java : il faut déclarer une méthode qui va être appelée sur l'événement « fin de l'Activity MainActivity » : surcharger une méthode onActivityResult()

Pour ce faire dans MainActivity.java, en dehors de toute méthode, Ctrl + O

taper le début de onActivityResult et Tab quand trouvé.

La bonne signature et squelette s'écrivent automatiquement.

Voici le contenu complet :

```
@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if (requestCode == ACTIVITY_CODE && resultCode == RESULT_OK)
    {
        int score = data.getIntExtra(MainActivity.BUNDLE_EXTRA_SCORE, 0);
        Log.i("main", "Le score est " + score);
        TextView text = findViewById(R.id.textView);
        text.setText(getText(R.string.id_welcome) + " Score de " + score);
    }
}
```


10.2.9 Utiliser les préférences d'une application

Une API Android permet de sauvegarder dans un fichier xml des données propres et privées à une application.

Le code pour écrire une préférence est :

```
SharedPreferences preferences = getPreferences(MODE_PRIVATE);  
SharedPreferences.Editor editor = preferences.edit();  
editor.putString("firstname", mUser.getFirstName());  
editor.apply();
```

Le code pour lire une préférence est :

```
String firstname = getPreferences(MODE_PRIVATE).getString("firstname", null);
```

Grâce à la section précédente, vous devez être en mesure de stocker et récupérer le prénom de l'utilisateur, tout cela dans la *MainActivity*.

Exercice

Je vous propose de mettre à jour le code de la *MainActivity* de la façon suivante :

- Lorsque l'activité démarre, vérifiez si une partie a déjà eu lieu (indice : si tel est le cas, un prénom et un score doivent avoir été enregistrés dans les préférences ;
- Récupérez ces valeurs et mettez à jour le texte d'accueil, en saluant la personne et en lui rappelant son précédent score ;
- Valorisez également le champ de saisie avec le prénom de la personne, afin d'éviter qu'elle ait à le saisir de nouveau. N'oubliez pas de positionner le curseur à la fin du prénom !
- Pensez également à mettre à jour l'affichage et sauvegarder les préférences lorsqu'une partie vient de se terminer

10.2.10 Temporiser l'application

Pour l'instant lorsque le Toast Correct ou Incorrect s'affiche, la question suivante s'affiche et l'appui sur les touches est possible.

Nous souhaitons calmer le jeu lorsque le Toast est affiché:

- attendre la fin pour poser la question suivante
- interdire l'appui sur les boutons

Modifier le code de GameActivity.java avec les aides suivantes :

Pour temporiser, c'est à dire demander à l'OS de suspendre un traitement, le code suivant peut être employé :

```
new Handler().postDelayed(new Runnable() {
    @Override
    public void run() {
        // If this is the last question, ends the game.
        // Else, display the next question.
    }
}, 2000); // LENGTH_SHORT is usually 2 second long
```

Pour empêcher l'appui sur les touches, code suivant :

```
@Override
public boolean dispatchTouchEvent(MotionEvent ev) {
    return touchEnable && super.dispatchTouchEvent(ev);
}
```

Dans Android Studio, Ctrl + O puis le début de dispatchTouchEvent pour le squelette de la méthode.

Quand dispatchTouchEvent renvoie false, les touches sont inhibées.

touchEnable est un attribut boolean qui gère les moments où il faut permettre/interdire l'appui sur les touches.

Tester.

10.2.11 Maîtriser les Activities

Si l'on change de mode paysage/portrait de la page 2 le comportement peut être étrange :

- changement de question
- le score est mal calculé
- le retour à la page 1 échoue
- ...

Ceci est dû au fait que le changement de mode fait que l'on recrée l'Activity !

Voir <https://openclassrooms.com/fr/courses/4517166-developpez-votre-premiere-application-android/4586901-comprenez-le-cycle-de-vie-dune-activite>

et le site

<https://developer.android.com/guide/components/activities/activity-lifecycle.html#java>

pour une explication.

Pour implémenter ceci : du courage !

Dans GameActivity.java :

- ajouter la méthode onSaveInstanceState() et mettre dans le Bundle les éléments à sauvegarder. Voici un exemple

```
@Override
protected void onSaveInstanceState(Bundle outState) {
    outState.putSerializable("questions", questionBank);
    outState.putInt("nbQuestions", nbQuestions);
    outState.putInt("score", score);
    super.onSaveInstanceState(outState);
}
```

- dans le onCreate existant adapter le code pour récupérer ces valeurs sauvegardées. Voici un exemple de code :

```
if (savedInstanceState != null)
{
    questionBank = (QuestionBank) savedInstanceState.getSerializable("questions");
    nbQuestions = savedInstanceState.getInt("nbQuestions");
    questionBank.decrementIndex();
    score = savedInstanceState.getInt("score");
}
else {
    questionBank = generateQuestions();
    nbQuestions = questionBank.getNBQuestions() ;
}
```

Le code ci-avant serialise l'objet de classe QuestionBank . Faut indiquer à cette classe qu'elle implémente l'interface java.io.Serializable

Tester

Section 11 Créer un compte Google Play

11.1 But

- Fournir des informations pour créer un compte
- Enregistrer notre application

11.2 informations

Se rendre sur le site

Rechercher sur le Web « Console Google Play » et aller sur le site

Se logger avec une adresse Google.



Cet enregistrement est payant et vaut 25\$ par an.

Après le paiement l'accès est possible à la console Google Play

Choisir « Créer une application »

Fournir un titre et description.

Fournir des images qui caractérisent l'application. Pour ce faire Visual Studio offre une aide :

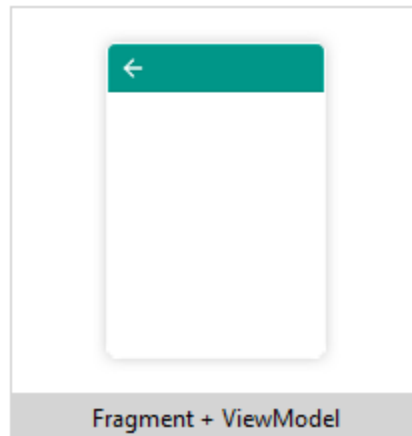
Lancer l'application dans le simulateur.



Sur Visual Studio, choisir Outils > Android > Désigner la ligne du simulateur puis sur l'appareil photo.

La console Google Play permet de gérer la version Bêta, Alpha, et production.

Créer un nouveau projet de type



Nommer LeLab2Minimum SDK Android6,0 Marshmallow