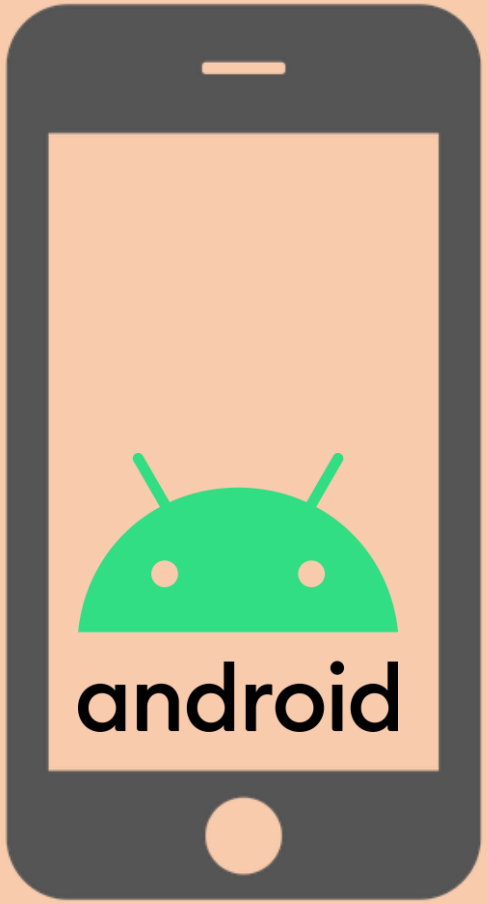
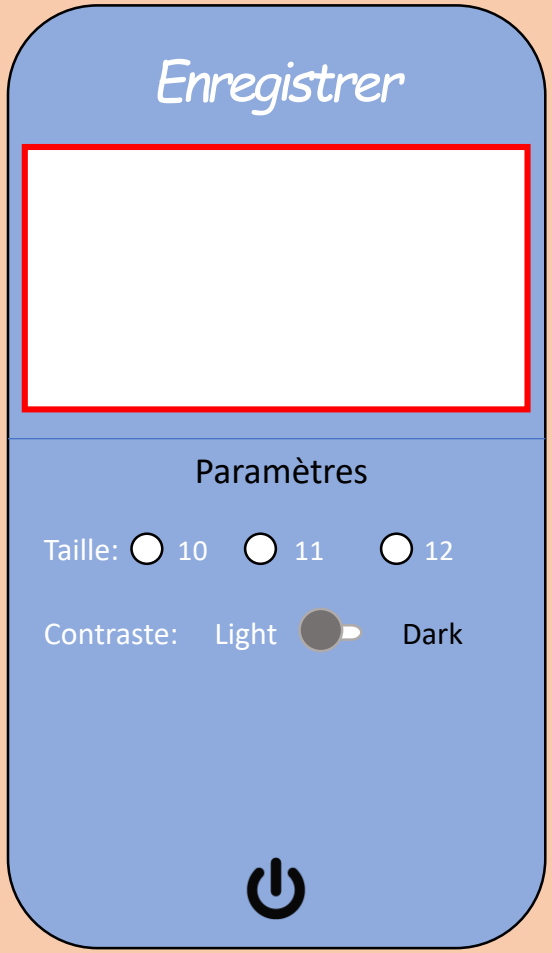
Quatrième exemple



La maquette: préparation des boutons

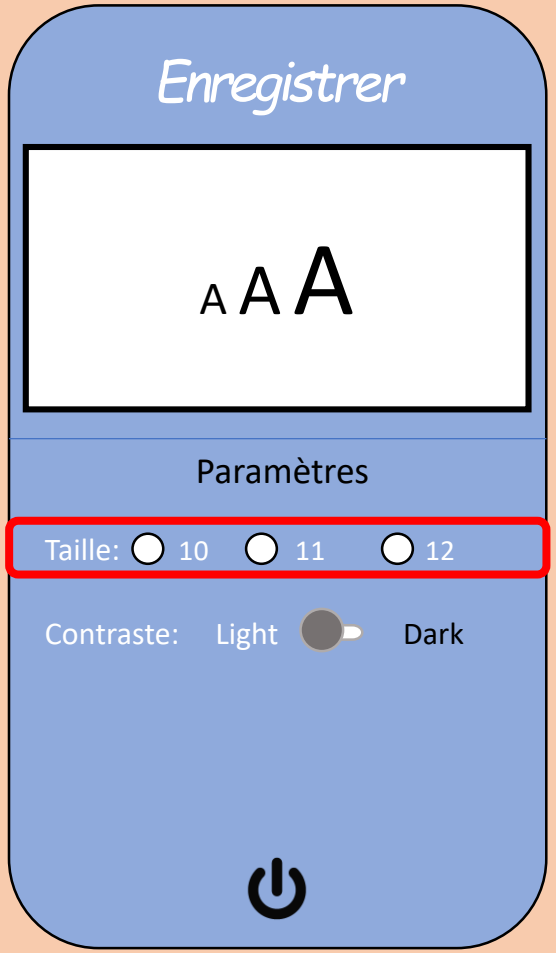


1. Utiliser un éditeur de texte:

Layout (XML):
- Utiliser un éditeur de texte qui permet la saisie d'une chaîne de caractères
- <EditTex/>

Programme(Kotlin):
- Créer la variable « private lateinit var nom: EditText *=> Type* »
- Initialiser la variable « findViewById(R.id.***this.nom***) »
- Maintenant on peut appliquer des modifications sur ce texte dans les différentes fonctions de toute la classe.

La maquette: préparation des boutons



1. Modifier la taille du texte:

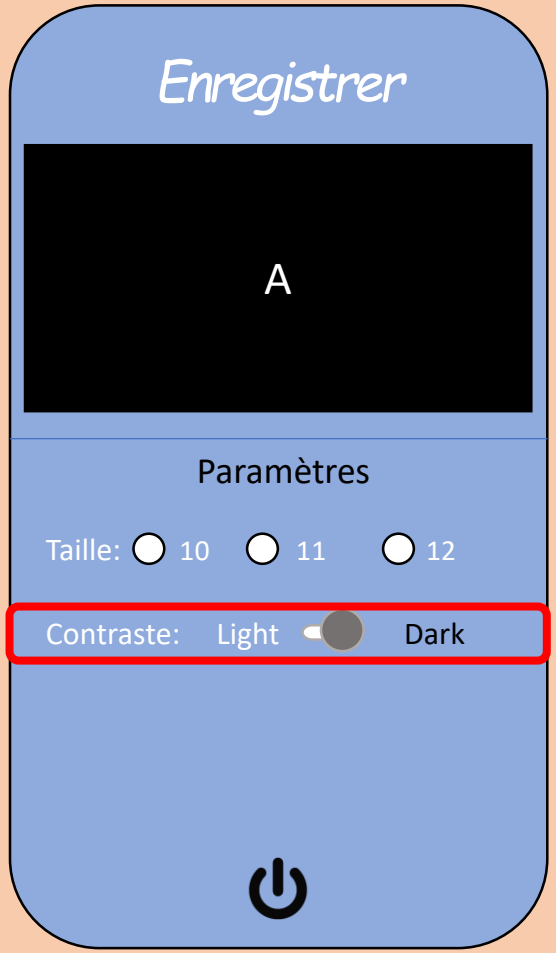
Layout (XML):

- Utiliser des boutons radios pour appliquer un seul choix
- <RadioGroup> <RadioButton/> … </RadioGroup>

Programme(Kotlin):

- Créer la variable « private lateinit var nom: RadioGroup *=> Type* »
- Initialiser la variable « findViewById(R.id.***this.nom***) »
- Listener -> pour le RadioGroup « .setOnChekedChangeListener{radiogroup, i ->} » i est un entier, il est l'identifiant du radio bouton sélectionné dans le RadioGroup.
- En fonction du bouton radio sélectionné l'on applique une taille au texte, donc on programme une condition
- Pour appliquer une taille au texte « ***this.text***.setTextSize(TypedValue.COMPLEX_UNIT_PT,10F) »

La maquette: préparation des boutons



2. Modifier le contraste du texte:

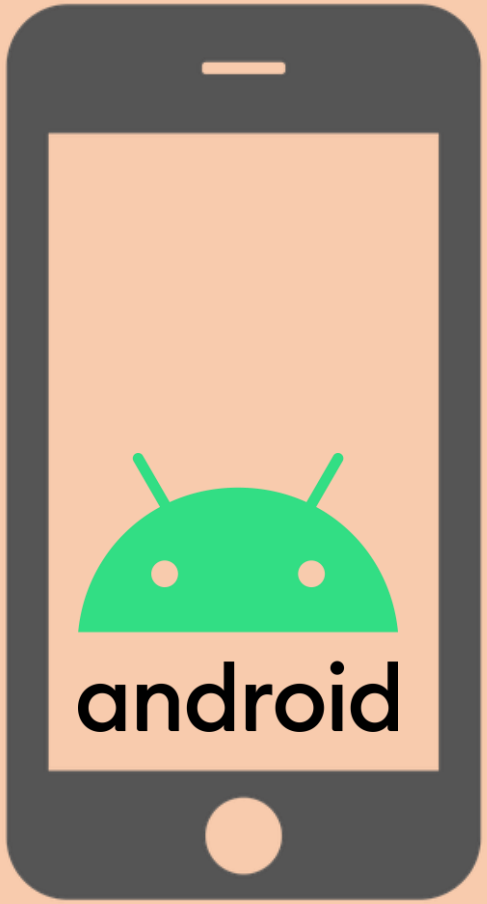
Layout (XML):

- Utiliser un bouton switch pour choisir parmi deux états
- <Switch/>

Programme(Kotlin):

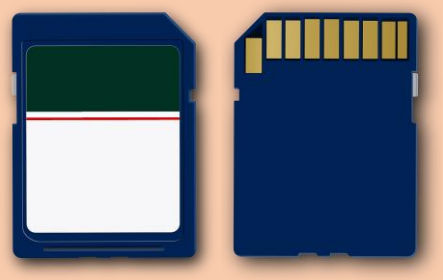
- Créer la variable « private lateinit var nom: Switch *=> Type* »
- Initialiser la variable « findViewById(R.id.***this.nom***) »
- Listener -> pour le Switch
  « .setOnChekedChangeListener{compoundButton, i ->} » i est un booléen il permet de savoir si le switch est activé ou non
- Pour appliquer un contraste il s'agit de modifier de couleurs: celle du texte
  « .setTextColor(***Color.???***) »et celle de l'arrière plan
  « setBackgroundColor(***Color.???***) »

Quatrième cours



Stockage des données

Une application enregistre et charge ses données depuis plusieurs sources de stockage:

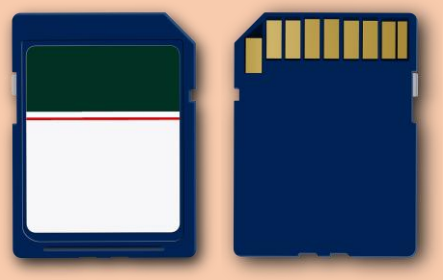


Où stocker les données ?

Images par Clker-Free-Vector-Images de Pixabay

Stockage des données

Une application enregistre et charge ses données depuis plusieurs sources de stockage:



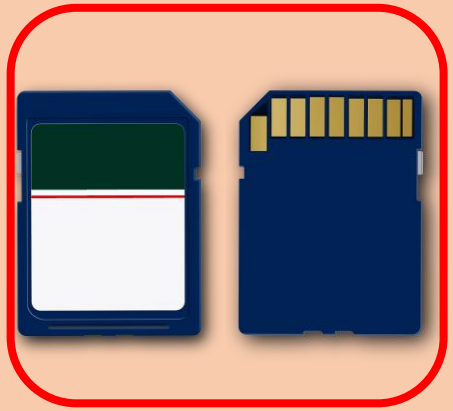
La mémoire **interne** (paramétrages, cache, mots de passe, etc.):

a. Espace de stockage limité
b. Plus sécurisé -> les utilisateurs n'y ont pas accès
c. Toujours accessible pour l'application et uniquement par l'application
d. Données effacées lors de la désinstallation de l'application

Images par Clker-Free-Vector-Images de Pixabay

Stockage des données

Une application enregistre et charge ses données depuis plusieurs sources de stockage:



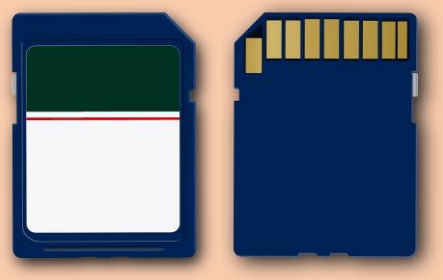
La mémoire **externe** (données à partager, volumineuses, à conserver, etc.):

a. Espace de stockage limité pour les smartphones + illimité (SD Card, USB Key)
b. Pas toujours accessible à l'application, données partageable ou non avec d'autres applications (Autorisations)
c. Nécessite des autorisations pour accéder aux données

Images par Clker-Free-Vector-Images de Pixabay

Stockage des données

Une application enregistre et charge ses données depuis plusieurs sources de stockage:



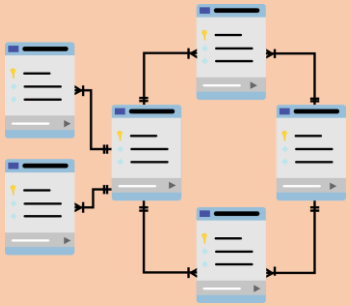
Un **serveur** (tous types de données):

a. Nécessite une connexion + autorisations (identification, ect.)

Images par Clker-Free-Vector-Images de Pixabay

La forme des données

Une application enregistre ses données de plusieurs façons selon leurs natures:

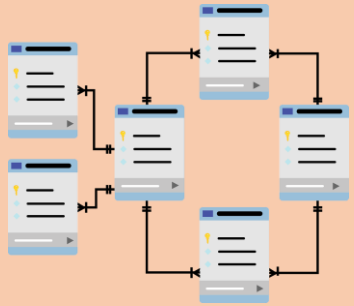


Sous quelle forme enregistrer les données ?

Images par OpenClipart-Vectors et OpenIcons et mcmurryjulie de Pixabay

La forme des données

Une application enregistre ses données de plusieurs façons selon leurs natures:



Les **préférences** (paramètres de l'application, etc.):

a. Utilise une clé pour récupérer une valeur
b. Sont enregistrer dans un ou plusieurs fichiers selon les besoins
c. Sont enregistrés dans la mémoire interne de l'application
d. Pour des données primitives et courtes

Images par OpenClipart-Vectors et OpenIcons et mcmurryjulie de Pixabay

Les préférences

L'API SharedPreference( ):

Importation « android.content.SharedPreferences »

- Un objet qui permet la création de pairs clés/valeurs choisies
- L'objet représente un fichier qui enregistres ces préférences avec un nom
- SharedPreference( ): https://developer.android.com/training/data-storage/shared-preferences?hl=fr

Editer les préférences partagées

L’objet Editor simplifie la gestion des données des préférences partagées via ses méthodes explicites:

L’on doit appliquer les modifications pour les récupérer:

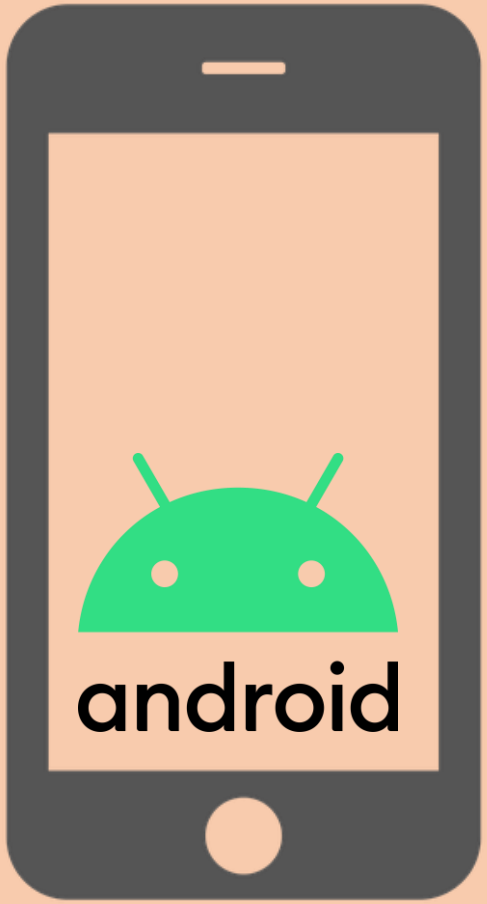
« votreEditeur.apply( ) »

L’on peut ajouter des données de valeurs primitives en spécifiant leurs clés (Chaîne de caractères):

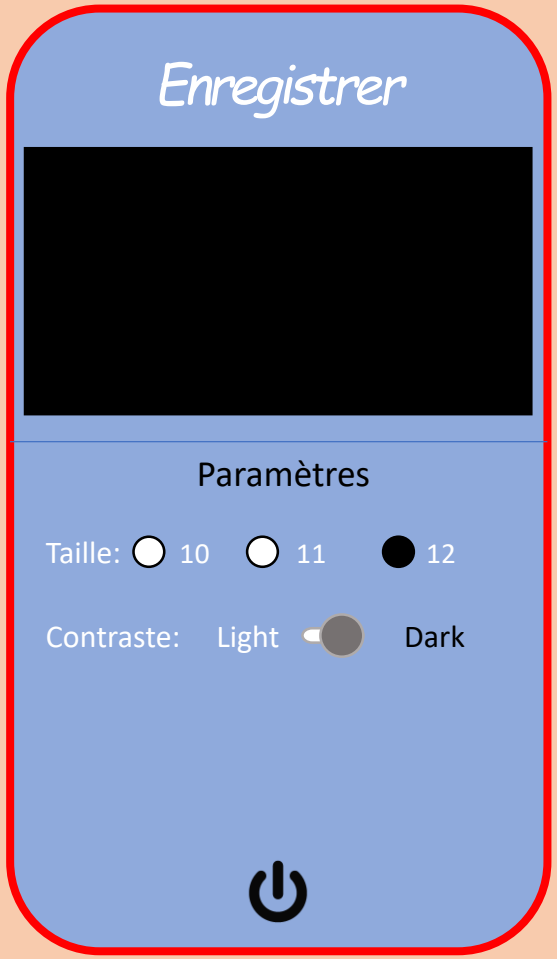
« putInt(‘’Clée’’, Int) »
« putFloat(‘’Clée’’, Float) »
« putBoolean(‘’Clée’’, Boolean) »
« putString(‘’Clée’’, String) »
…

Editor( ): https://developer.android.com/reference/android/content/SharedPreferences.Editor

Quatrième exemple



La maquette: enregistrer les paramètres



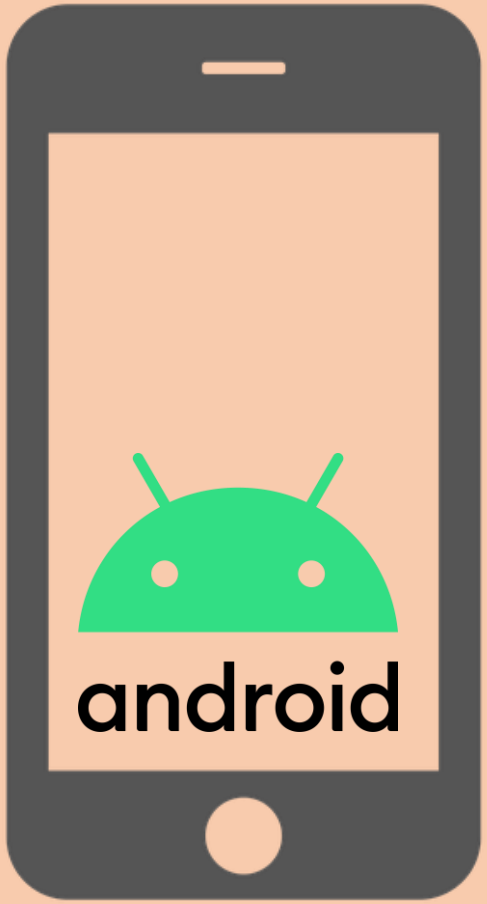
1. **<u>Le programme (fichier kotlin):</u>**

   - Déclarer des variables pour la classe de type SharedPreference et Editor
   - Initialiser les préférences avec un nom pour le fichier et un Mode « ***this.sharedPreferences*** = getSharedPreferences(''Preferences'', MODE_PRIVATE) »
   - Initialiser l'éditeur pour qu'il puisse modifier les préférences « ***this.editor*** = ***this.sharedPreferences***.edit() »

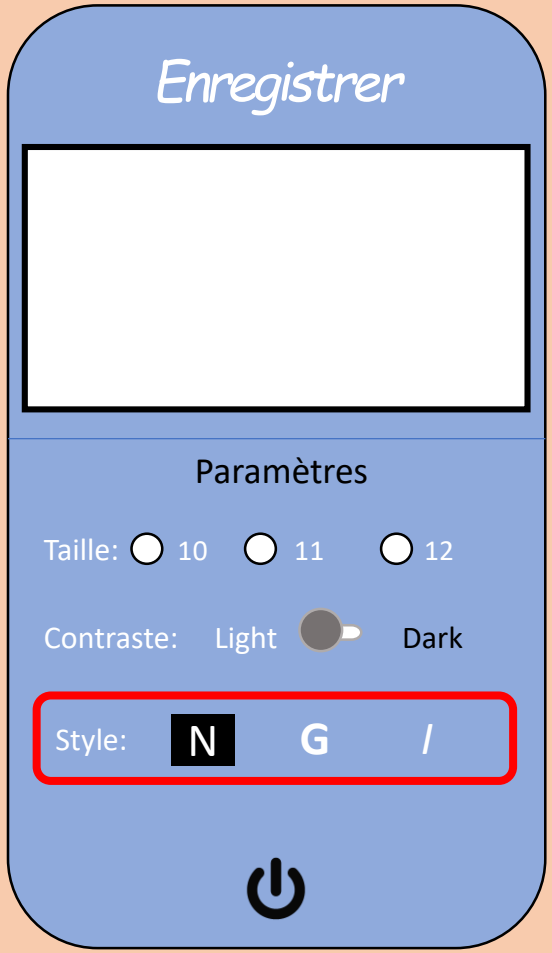
   - Dans les deux fonctions toSize( ) et toContrast( ) enregistrer les préférences:
     « ***this.editor.apply*** {put???(''clé'', valeur de type ???)}.apply() »
     ??? => Int, Float, Boolean, String, etc.

   - Lorsque les préférences sont enregistrées ou doit les charger à l'ouverture de l'application
     Créer une fonction toLoad( ) pour charger les préférences enregistrées
     Dans cette fonction on définit les états des boutons et du texte
     « ***this.sharedPreferences.get???***(''clé'', valeur par défaut au cas où) »

Quatrième exercice



La maquette: ajouter un style au texte



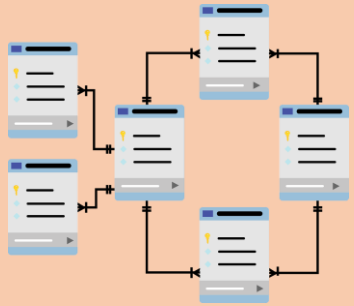
1. ***Layout (fichier xml)***:
   - *Ajouter un nouveau <LinearLayout> </LinearLayout> avec un identifiant « style »*
   - *Lui attribuer le sous-titre « Style »*
   - *Lui ajouter trois <Button> pour les options « Normal », « Gras », et « Italique »*
   - *Faire en sorte que le <LinearLayout> soit aligné avec le précédent identifié par « contrate »*
   - *Corriger <ImageButton> pour qu'il ne soit plus aligné avec « contraste » mais avec « style »*

2. **Programme (fichier kootlin):**
   - **Déclarer les variables et créer une nouvelle fonction en prenant exemple sur les autres boutons du programme.**
   - **utiliser « .setTypeFace(null, *Typeface.???*) » pour appliquer un style au texte. Les types sont: « Typeface.Normal », « Typeface.BOLD », « Typeface.ITALIC »**

3. **Programme (fichier kootlin):**
   - Enregistrer le dernier style choisit pour l'éditeur dans le SharedPreferences ( ), grâce à l'outil Editeur « ***editeurName***.apply {…}.apply()
   - Charger dans la fonction toLoad( ) le style du texte pour qu'il soit appliqué dès l'ouverture de l'application
   - Faire en sorte que le bouton précédemment sélectionné change de couleur d'arrière plan « .setBackgroundColor(***Color.???***) »

La forme des données

Une application enregistre ses données de plusieurs façons selon leurs natures:



Les **fichiers** (données volumineuses, à conserver, pour plusieurs applications, etc.):

a. Peuvent être stockées dans la mémoire interne et externe de la tablette
b. Besoin d'autorisations pour les fichiers externes
c. Importation des classes File( ), FileOutputStream( ), FileInputStream( )
d. Utiliser un buffer pour lire et écrire les données

Images par OpenClipart-Vectors et OpenIcons et mcmurryjulie de Pixabay

Stocker des fichiers

Où stocker des fichiers ?

On peu les stocker dans les mémoires internes ou externes du smartphones

Comment stocker des fichiers?

Cela dépende de l'emplacement de la mémoire

Pour la mémoire interne:

classe FileOutpuStream( ):
https://developer.android.com/reference/java/io/FileOutputStream

Classe FileInputStream( ):
https://developer.android.com/reference/java/io/FileInputStream

BufferReader( ):
https://developer.android.com/reference/java/io/BufferedReader

Images par OpenClipart-Vectors et OpenIcons et mcmurryjulie de Pixabay

Stocker des fichiers

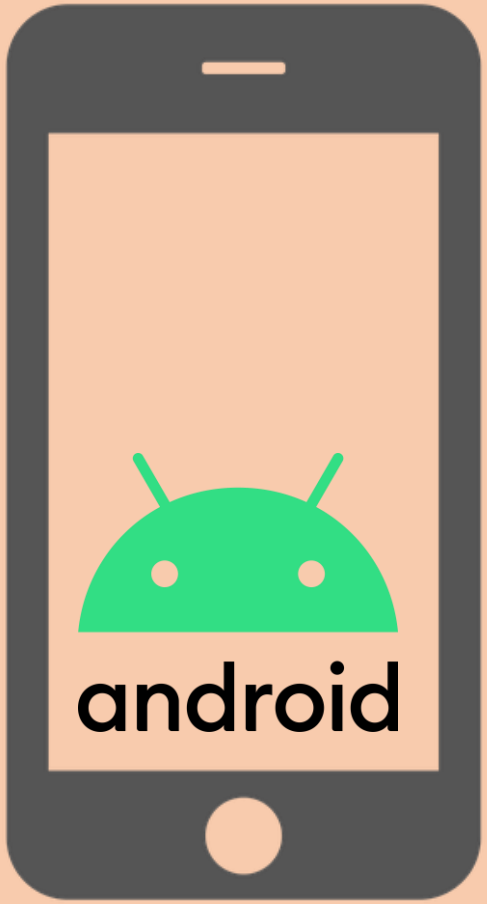
Comment stocker des fichiers?
Cela dépende de l'emplacement de la mémoire

Pour la mémoire externe:

- Vérifier que l'emplacement est disponible
- Obtenir les autorisations pour lire, modifier et executer les documents
- class File(): https://developer.android.com/reference/java/io/File
- Context( ) -> getExternalFilesDir(«null»): https://developer.android.com/reference/android/content/Context

Images par OpenClipart-Vectors et OpenIcons et mcmurryjulie de Pixabay

Cinquième exemple



La maquette: stocker les textes

Objectif: Stocker un fichier dans les mémoires interne et externe du smartphone

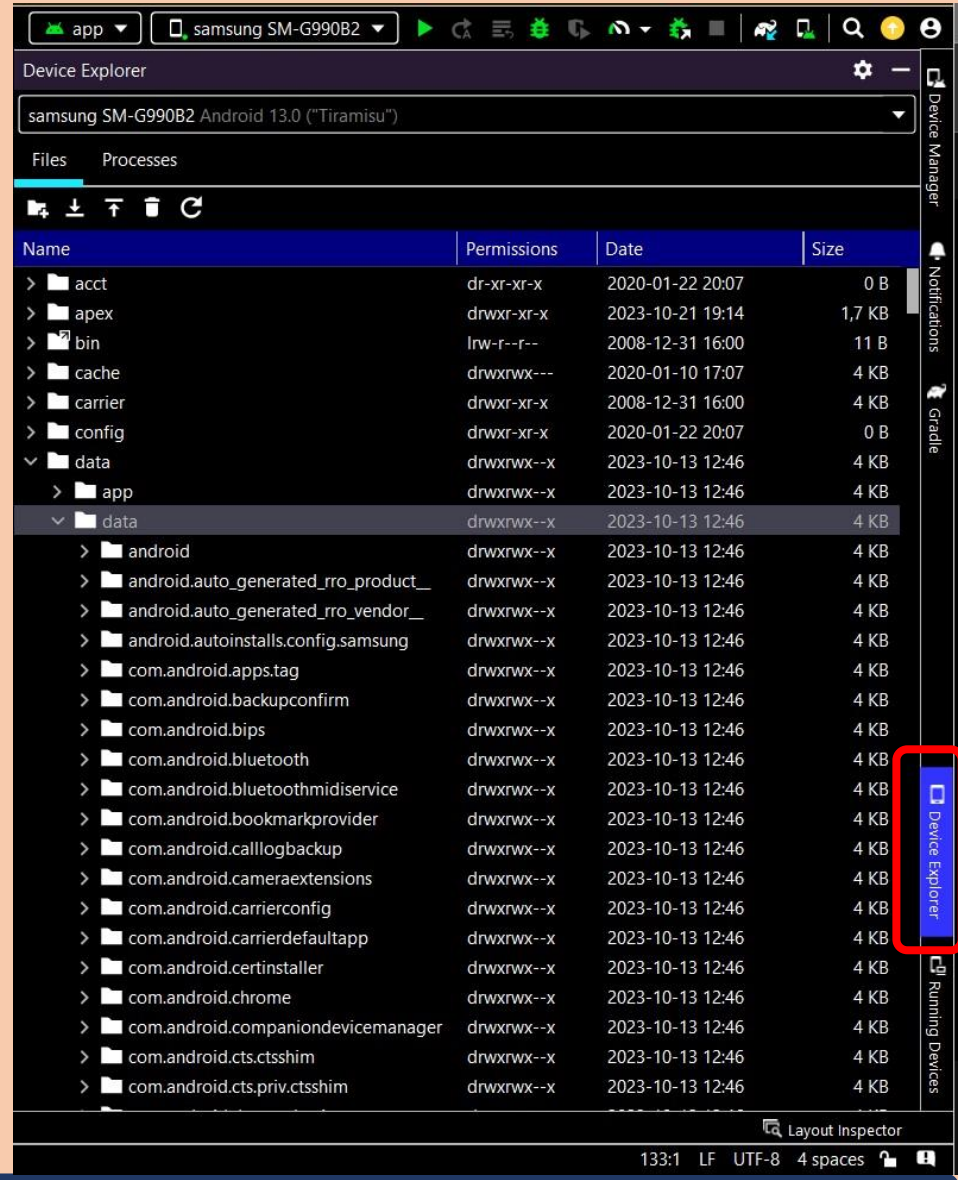
- Créer un fichier au nom du titre
- Remplir le fichier avec une chaîne de caractères
- Sauvegarder le fichier
- Supprimer le fichier
- Ouvrir le fichier

Device explorer la mémoire interne

Le Device explorer vous permet de naviguer dans les différents fichiers de votre smartphone pour visualiser les données:

Accéder aux données enregistrées par l'application que vous développée:

Pour acceder à la mémoire interne qui stocke le plus souvent: les préférences (SharedPreferences) , les fichiers sensibles, il faut ouvrir le dossier « data/data/com.exemple.exemple.***application*** »



Device explorer mémoire externe

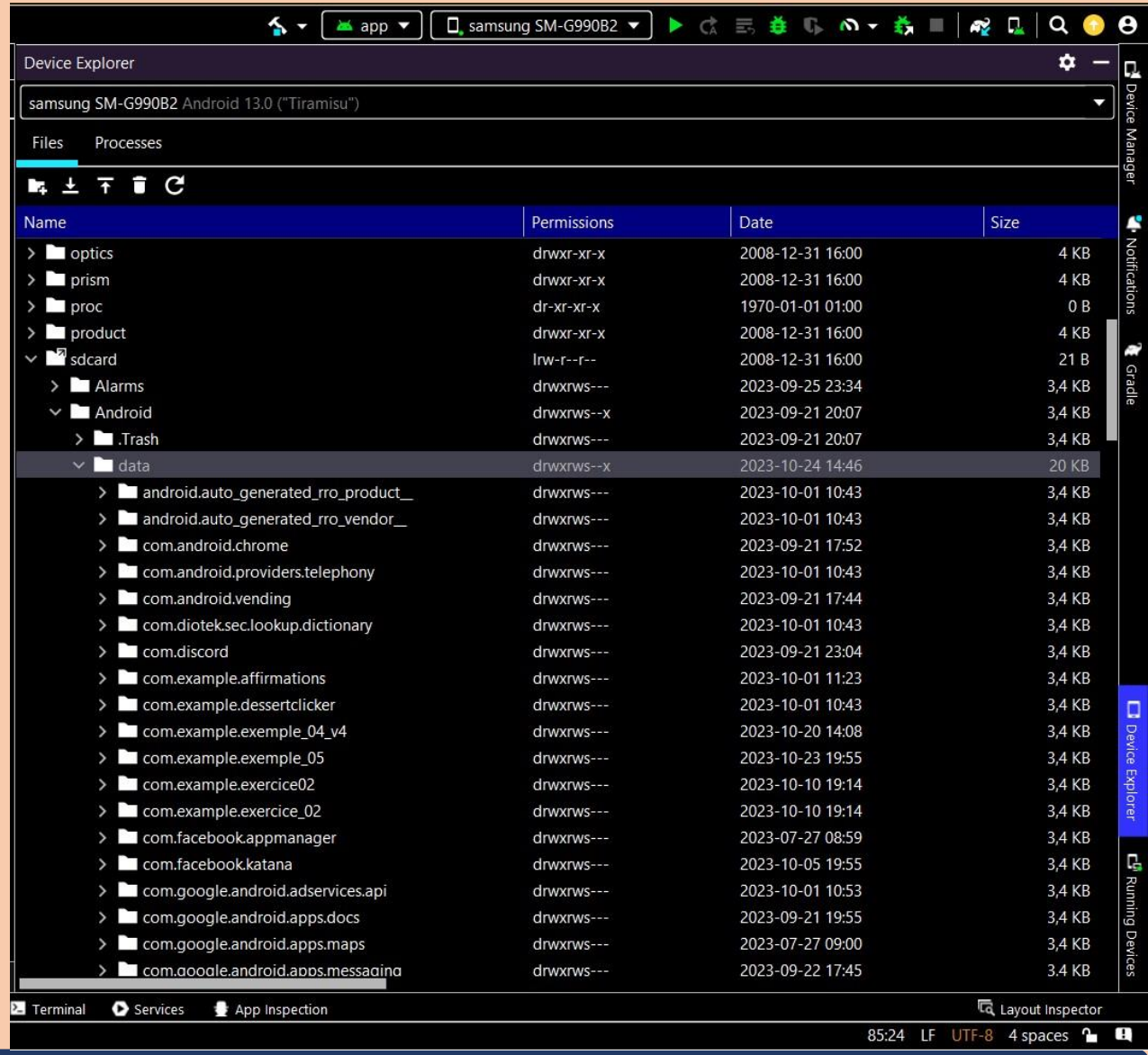
Pour visualiser les fichiers stockées dans la mémoire externe:

les deux chemins possible
« storage/self/data/com.exemple.exemple.***application*** » et
« sdCard/self/data/com.exemple.exemple.***application*** »

L'on peut donc enregistrer et récupérer des données (fichiers, images, vidéos etc.) persistantes (qui ne se suppriment pas après désinstallation de l'application) dans des dossiers de la mémoire externe pour qu'elles puissent être accessible aux autres applications.

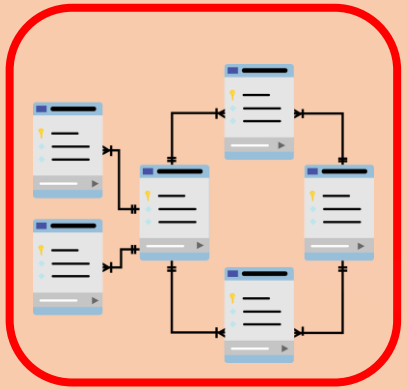
On ne peut pas ouvrir et visualiser les fichiers des autres applications (Instagram etc.) si elles ne sont pas partagées dans les dossiers Medias.

On peut vérifier que la machine virtuelle est étaeinte si aucun fichiers n'apparaît dans le Device explorer



La forme des données

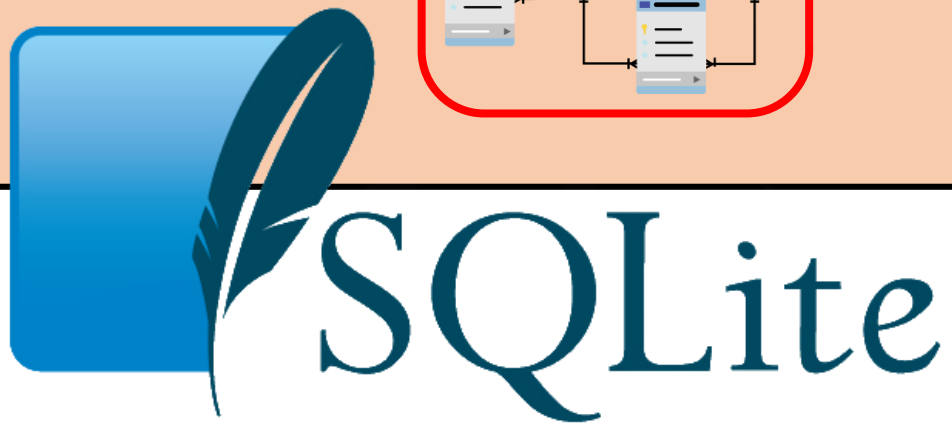
Une application enregistre ses données de plusieurs façons selon leurs natures:



Les **données structurées**:

a. Différentes types de données - dynamiques
b. Distribution des données dans une interfaces (avec un model)

Nous ne verrons pas les données structurées lors de ce cours.



Images par OpenClipart-Vectors et OpenIcons et mcmurryjulie de Pixabay