Travaux Dirigés et Pratiques de Programmation Android n° 1

Premiers pas

Les objectifs de ce TD sont :

- L'installation et la configuration des outils qui vont être utilisés tout au long du module ;
- La découverte d'une première application Android ;
- Quelques rappels sur Java.

Vous êtes libres d'utiliser l'environnement de votre choix, sous réserve d'être à l'aise avec. Les TDs sont rédigés pour l'utilisation d'Android Studio.

Si vous souhaitez travailler sur les machines de l'IUT, allez au paragraphe (a) puis allez directement au paragraphe (c). Si vous souhaitez travailler sur votre machine personnelle, allez directement au paragraphe (b) puis allez au paragraphe (c).

a) Utilisation des machines de l'IUT

Android Studio est déjà installé sur les machines de l'IUT, sous Windows uniquement.

1. Le lancer.
2. Choisir « I do not have a previous version of Studio [...] ».
3. Avancer dans le wizard. Choisir une installation « Custom », avec comme thème « IntelliJ ». Sélectionner tous les composants à installer.
4. À la fin du wizard, choisir « Configure/SDK Manager ». Quelles sont les versions de l'API Android qui sont installées ?

b) Installation sur une machine personnelle

1. Télécharger Android Studio (https://developer.android.com/sdk/index.html) pour votre système d'exploitation.
   - Sous Windows ou MacOS, double-cliquer sur le fichier téléchargé.
   - Sous Linux, décompresser l'archive, puis lancer le studio en utilisant le script `android-studio/bin/studio.sh`. (Remarque : on pourra utiliser toujours ce script lors des autres TD pour lancer le studio.)
2. Choisir « I do not have a previous version of Studio [...] ».
3. Avancer dans le wizard. Choisir une installation « Custom », avec comme thème IntelliJ. Sélectionner tous les composants à installer.
4. À la fin du wizard, choisir « Configure/SDK Manager ». Par défaut, l'API installée est la plus récence, l'API 23. Nous allons également installer l'API 15 afin de pouvoir installer nos applications sur un maximum de plateformes aujourd'hui utilisées, et en particulier nos tablettes :
   - Dans « API 23 », tout décocher sauf « ARM EABI [...] » ;
   - Dans « API 15 », cocher « SDK Platform », « ARM EABI [...] », « Google APIs » et « Sources for Android SDK » ;
   - Dans « Extras », cocher « Android Support Library » et « Google Play Services » ;
   - Cliquer sur « Install *x* packages... » ;
   - Pendant que ça s'installe, lire le paragraphe c) pour s'occuper.

c) Environnement utilisé

Voici quelques questions et remarques sur l'environnement utilisé.

1. Définir le JRE et le JDK. Quelle version du JDK est disponible sur votre machine ?
2. Le SDK, pour « Software Development Kit », est une API Java proposant un ensemble de classes pour le développement d'applications Android. En constante évolution, il existe différentes versions, chacune étant compatible avec la précédente (pour les versions actuelles). Une application grand public doit être compatible avec les versions les plus utilisées au moment de sa diffusion, afin d'être utilisable par le plus grand nombre ; c'est pourquoi nous allons, en plus d'une API récente (22 ou 23, selon la machine que vous utilisez), en utiliser une plus ancienne.
3. Rechercher la version d'Android installée sur la tablette, et la version maximale du SDK qu'elle supporte.

Première application

1. Quitter le « SDK Manager » et revenir en arrière.
2. Choisir « Start a new Android Studio Project ». Choisir le nom que vous souhaitez lui donner (par la suite, nous l'appellerons « TD1 ») et l'endroit où vous souhaitez l'enregistrer (de manière à pouvoir le retrouver : si vous travaillez sur les machines de l'IUT, enregistrez-le sur votre compte LDAP (Z:)).
3. Choisir comme cible « Phone and Tablet », avec l'API 15 comme SDK minimum.
4. Choisir « Blank Activity ». Là encore, vous pouvez lui donner le nom que vous souhaitez ; par exemple, le même nom que celui de l'application, suivi de « Activity » (dans notre cas, « TD1Activity »).
5. Si vous disposez d'une tablette, installez cette application (pour l'instant basique) dessus :
   - Allumer et brancher la tablette.
   - Dans le studio, cliquer sur le triangle vert (« Run 'app' ») et choisir la tablette dans « Choose a running device ». L'application s'installe alors et se lance sur la tablette ; elle est également dorénavant disponible dans le menu des applications.

Création d'un émulateur

Un émulateur sert à émuler un appareil Android particulier, afin de développer une application pour divers types d'appareil (téléphone ou tablettes, avec différentes tailles d'écran...) sans pour autant avoir besoin de tous les posséder... !

Dans le cadre de ce module, l'émulateur pourra vous servir également à travailler lorsque vous n'avez pas accès aux tablettes. Pour cela, nous allons créer un émulateur dont ayant des caractéristiques similaires à celles de la tablette.

1. Cliquer sur l'icône avec un rectangle violet « AVD Manager » (pour « Android Virtual Device ») ;
2. Choisir « Create Virtual Device... », puis choisir parmi les tablettes l'architecture prédéfinie la plus proche de votre tablette (ou à défaut, un Nexus 10), avec une architecture ARM, l'API 15 et Android 4.0.3. Lui donner le nom que vous souhaitez ;
3. Lancer l'émulateur ;
4. Installer l'application comme précédemment (triangle vert), mais en choisissant cette fois-ci l'émulateur.

Découverte de l'application

Regarder ce que fait l'application. Vous l'aurez constaté, pour l'instant, elle se contente d'afficher « Hello world! », et propose également un menu avec l'option "Settings" n'ayant aucun effet. Nous allons présenter les différents composants de cette application, qui feront ensuite chacun l'objet de différents TDs.

1. Dans le menu à gauche du studio, déplier le répertoire « app ». Il contient trois sous-répertoires:
   - le répertoire `manifests` contient les fichiers décrivant l'architecture de l'application ;

   - le répertoire `java` contient les fichiers source Java ;
   - le répertoire `res` contient tout ce qui a trait à l'interface graphique de l'application.
2. Consulter le fichier `AndroidManifest.xml` du répertoire `manifests`. Il décrit donc l'architecture de l'application : son nom, son thèmes, les activités qu'elle contient (il n'y en a pour l'instant qu'une), ... Ce fichier doit être édité lorsqu'on ajoute de nouvelles activités, à la main ou automatiquement via l'utilisation des fonctionalités d'Android Studio.
3. Consulter le fichier `TD1Activity.java` du répertoire `java`. Il s'agit d'une classe Java implantant l'activité, contenant trois méthodes :
   - La méthode `onCreate` est appelée à la création de l'activité. Un de ses rôles est de démarrer l'interface graphique, via l'appel à la méthode `setContentView` ;
   - Les deux autres méthodes sont appelées respectivement à la création du menu (`onCreateOptionsMenu`), en démarrant l'interface du menu, et lorsque l'utilisateur choisit une option du menu (`onOptionsItemSelected`), en effectuant une action en fonction de l'élément sélectionné (pour l'instant, il n'y a qu'un seul élément, repéré par l'identifiant `R.id.action_settings`, et l'action effectuée se contente de renvoyer `true`).
4. Ouvrir le fichier `activity_td1.xml` du répertoire `res/layout`. Il décrit l'interface graphique de l'activité. Android Studio propose deux vues pour lire ce fichier :
   - l'onglet « Text » affiche simplement le contenu du fichier, à savoir du XML décrivant les diverses couches de l'interface, ainsi qu'une prévisualisation du résultat (pour laquelle on peut choisir différents appareils, notamment notre émulateur) ;
   - l'onglet « Design » affiche également la prévisualisation du résultat et permet facilement d'ajouter de nouveaux éléments par glisser-déposer.

   Regarder le contenu XML (fichier `content_td1.xml`) : l'interface contient une seule couche de type `RelativeLayout`, elle-même contenant une zone de texte `TextView` contenant la chaîne de caractères « Hello world! ». En passant le curseur sur cette chaîne, on constate qu'elle est en réalité définie dans la variable `@string/hello_world`, que nous allons voir maintenant.
5. Cette variable est définie dans le fichier `strings.xml` du répertoire `res/values`, ainsi que le nom de l'application, et une variable contenant l'unique élément du menu actuel. Le fait de définir ainsi des variables contenant les chaînes de caractères apparaissant dans l'interface permet de gérer facilement les différentes localisations de l'application (version anglaise, française, ... etc), comme nous le verrons par la suite.
6. Enfin, consulter le fichier `menu_td1.xml` du répertoire `res/menu`. De manière similaire au fichier `activity_td1.xml`, il décrit les éléments du menu : il contient pour l'instant un seul "item", ayant pour nom le contenu de la variable `@string/action_settings`, et pour identifiant `action_settings` que nous avons vu dans la méthode `onOptionsItemSelected`.

Un peu de code... enfin !

Si vous avez suivi le précédent paragraphe (et comme vous connaissez Java), vous devriez pouvoir faire les modifications suivantes. La modification précédée de « * » est un peu plus subtile, et nous y reviendrons lors de TDs suivants.

1. Modifier l'application de manière à ce qu'elle affiche « This is my first Android app. » au lieu de « Hello world! ».
2. Modifier l'application de manière à ce qu'elle affiche « Activity created » sur la sortie standard lorsque l'activité principale de l'application est créée. La sortie standard peut être lue dans l'onglet « 6: Android/logcat » du studio.
3. Modifier l'application de manière à ce qu'elle affiche « Settings selected » sur la sortie standard lorsque l'utilisateur choisit l'élément « Settings » du menu.
4. (*) En vous inspirant de ce qui existe déjà, ajouter un nouvel élément au menu appelé « Test », et faites en sorte que « Test selected » s'affiche sur la sortie standard lorsque l'utilisateur choisit cet élément.

Pour chaque modification, tester l'application sur la tablette ou l'émulateur.

Gestion de la localisation

Pour aller plus loin, voici comment on peut très facilement gérer plusieurs langues pour une même application, en ayant pris la précaution de définir toutes les chaînes de caractères de l'interface dans des variables du fichier `strings.xml` du répertoire `res/values`.

Par défaut, ce fichier décrit la locale en langue anglaise. Pour gérer la locale française, par exemple, il suffit de définir les mêmes variables dans un fichier également nommé `strings.xml`, mais dans le répertoire `res/values-fr`.

Pour créer simplement ce fichier avec Android Studio, procéder comme suit. Effectuer un clic droit sur `res/values/strings.xml` dans la colonne de gauche et choisir « New/Values resource file ». Comme nom, choisir « strings », puis prendre l'option « Locale/fr:/Any Region ».

Définir la version française des chaînes de caractères associées aux variables de `res/values/strings.xml`.

Installer et lancer l'application sur la tablette, et observer. Lors de l'installation, Android Studio choisit automatiquement une localisation en fonction des paramètres de la tablette.

Si vous utilisez un émulateur, il faut le mettre en français (dans les paramètres) et le redémarrer pour observer le changement de localisation.

Attention : lorsqu'on a plusieurs localisations, il ne faut pas oublier de gérer les variables pour chacune d'entre elles ! Il est souvent préférable de toujours faire une passe sur les fichiers de localisation à la fin, après chaque modification d'une application.