IUT - Departement Informatique

Android - 2014-2015

1ère séance - Familiarisation avec Android

Android TD1

Ce premier TD a pour but de vous familiariser avec l'environnement de développement d'Android.

1. **Création d'un projet.** Commencez par lancer l'Android Studio en tapant `asb` dans un terminal. Vous pouvez ensuite créer un nouveau projet : `File > New project`. Choisissez un nom pour votre application, un nom de paquetage (le nom par défaut peut convenir), ainsi que le répertoire dans lequel sera sauvegardé votre projet. Cliquez sur `Next`. Vous pouvez choisir la version d'Android cible pour votre application. C'est à double tranchant : plus la version est récente et plus l'API (ensemble des fonctionnalités dont vous disposez comme développeur) sera riche, au détriment du nombre de téléphones capables de faire tourner votre application (jetez à oeil au lien `Help me choose`). Cliquez sur `Next`. Choisissez `Blank Activity`, puis `Next` et enfin `Finish`.
2. **Si vous avez votre propre appareil.** Si vous avez un téléphone ou une tablette sous Android, connectez-la en USB et activez les options pour développeurs dessus (différemment selon les versions, c.f. Internet si vous ne trouvez pas). Lancez l'exécution en cliquant sur la flèche verte. Votre appareil devrait apparaître dans la rubrique "Choose a running device". Si vous ne voyez que des points d'interrogation, appelez-moi. Sinon c'est gagné, cliquez sur OK et cela devrait lancer l'application sur votre appareil.
3. **Si vous n'avez pas d'appareil sous Android.** Pas d'inquiétude, il existe un émulateur. Lancez l'exécution en cliquant sur la flèche verte. Puis, dans la rubrique "Launch emulator" cliquez sur "..." pour lancer le gestionnaire de périphérique virtuel. Cliquez sur `Create`, puis choisissez un nom (p.ex. "Telephone"), un Device (p.ex. "Nexus 4", relativement léger à émuler), une target (API Level 19 conseillée), Intel Atom x86, No skin, ... puis cliquez sur OK. Cliquez sur `Start`. Cela peut prendre un certain temps. Une fois l'émulateur lancé, vous pouvez fermer le gestionnaire et revenir dans la fenêtre modale précédente, où vous choisirez `Telephone` et cliquerez sur OK. Pas besoin de le démarrer à chaque fois. En fait, votre émulateur peut rester ouvert entre les exécution, chacune écrasant simplement la précédente. Malgré cela, vous vous entendrez bientôt dire "zut, j'ai fermé la fenêtre". Pour pourrez alors relancer l'émulateur déjà créé, directement, lors de la prochaine exécution.
4. **Arborescence du projet.** Après avoir exécuté votre application, prenez quelques minutes supplémentaires pour examiner les fichiers de l'arborescence et tenter d'en comprendre l'utilité. En particulier, rendez-vous dans le répertoire `res/layout` et regardez le contenu du fichier xml qui s'y trouve. Ce fichier décrit les composants graphiques de votre activité. Cette interface est composée d'un layout racine (dont le type `RelativeLayout` est un cas particulier de `ViewGroup`) dans lequel se trouve un champ de texte (dont le type `TextView` est un cas particulier de `View`). La valeur du texte lui-même est identifiée par la variable `@string/hello_world`. Trouvez le fichier qui définit cette variable et modifiez-la ! Regardez aussi le code source `java` dans le dossier `java` et tentez de comprendre comment se fait le lien entre le layout et votre application. Les méthodes `onCreateOptionsMenu()` et `onOptionsItemSelected()` ne sont pas essentielles pour le moment. Vous pouvez quand même tenter de comprendre de quoi il s'agit. D'ailleurs, y aurait-il un autre fichier de layout quelque part ? De quoi décrit-il les composants graphiques ? Où se trouve-t-il dans votre arborescence ?
5. **Tutoriel en ligne.** Vu qu'il existe un tutoriel en ligne très bien conçu (en anglais, cependant), nous allons suivre ce tutoriel pas à pas, en s'assurant que l'on comprend bien les concepts au fur et à mesure. Nous vous invitons donc à revenir à cette feuille de TD à chaque fois que vous atteignez le bas d'une page du tutoriel, pour faire le point (en français).

   Nous allons prendre le tutoriel en cours de route, car les premières pages étaient relatives à l'environnement Eclipse, qui laissera bientôt sa place à l'Android Studio que nous utilisons. Rendez-vous donc sur `http://developer.android.com/training/basics/firstapp/building-ui.html`. Dans cette troisième page vous apprenez à créer une interface graphique élémentaire, composée d'un layout qui contient une zone de texte et un bouton. Prenez le temps de comprendre ce que vous écrivez, il y a plusieurs subtilités comme par exemple le « + » dans `android:id="@+id/edit_message"`.

   →Une fois en bas de page, vous devez avoir pris connaissance des points suivants:

   (a) Android intègre de manière native plusieurs bonnes pratiques. C'est le cas pour la séparation du code et des diverses ressources, par exemple en incitant fortement le développeur à spécifier son interface graphique dans un fichier XML séparé. (Il reste tout à fait possible, et parfois nécessaire, de créer les composants graphiques directement depuis le code java.) C'est aussi le cas pour l'internationalisation, qui est explicitement supportée par le biais d'un

fichier `string.xml` dans lequel toutes vos chaînes de caractère doivent être regroupées. Si vous ne savez pas ce que l'internationalisation est, n'hésitez pas à demander à votre voisin (soit il saura, soit il aura un voisin qui soit saura, soit aura un voisin qui ...).

(b) Toutes les ressources (composants graphiques, chaînes de caractères, etc.) possèdent un nom associé à un identifiant unique de type entier. L'ajout du symbole + dans la déclaration d'un identifiant permet de générer ce dernier automatiquement. On peut ensuite récupérer un identifiant donné à partir de son nom depuis le code java (nous verrons cela plus tard) pour manipuler la ressource correspondante.

(c) Les propriétés de positionnement des composants sont nombreuses et non-triviales, mais il n'est pas essentiel de les maîtriser toutes.

6. Vous pouvez passer à la dernière page du tutoriel, qui vous expliquera comment associer un traitement particulier lorsqu'un bouton est pressé. Vous verrez aussi comment une activité peut lancer une autre activité en utilisant le mécanisme des *Intents* : sortes de directives émises par une activité à l'intention des autres. Attention, certaines consignes sont faites pour Eclipse, il faudra les adapter pour l'Android Studio que vous utilisez. Par exemple, pour créer une nouvelle activité, on peut cliquer droit sur `java` > `new` > Activity > Blank Activity, etc. Ignorez aussi les consignes relatives aux fragments (tout en restant curieux à leur égard).

   →Une fois en bas de page, vous devez avoir pris connaissance des points suivants:

   (a) On peut associer l'action d'un bouton pressé à une méthode dans le code source. Mais au fait, comment le système sait-il dans quelle activité cette méthode est définie ?

   (b) Une activité peut en lancer une autre en lui passant aussi des paramètres. Cette dernière peut renvoyer l'utilisateur à l'activité appelante, par programme (quelle ligne de votre code gère ça ?).

7. Autres questions :

   (a) La première activité continue-t-elle à s'exécuter lorsque la seconde démarre ? On vous a dit qu'Android était multitâche, voyez-vous une bonne raison pour que ça ne soit pas le cas ?

   (b) La première activité pourrait-elle envoyer un message au bout de 20 secondes à l'utilisateur pour lui dire qu'il passe trop de temps avec la seconde ?

   (c) La seconde activité devient-elle visible avant ou après que la méthode `onCreate()` ait été invoquée ?

8. Si vous avez terminé, nous vous conseillons le tutoriel suivant, qui permet d'ajouter une "Action Bar" à votre application.