Systèmes mobiles
Laboratoires : Informations générales

15.09.2020

Installation de l'environnement de développement

Vous pouvez télécharger et suivre les instructions d'installation sur la page du site *Android* : https://developer.android.com/studio/index.html. On prendra garde aux points suivants :

- Depuis 2015, *Eclipse* n'est plus supporté par *Google* comme environnement de développement ; il est nécessaire d'utiliser *Android Studio*, qui est basé sur la plate-forme *IntelliJ*.
- Depuis 2019, le langage *Kotlin* est devenu le langage prioritaire pour le développement sur *Android*. Avec *Java*, les deux langages coexistent et il est possible de développer avec l'un, l'autre ou un mélange des deux.

Donc attention aux anciens tutoriels encore disponibles sur le web.

Les outils

On désire ici mentionner quelques-uns des principaux outils à disposition dans l'environnement *Android*. En plus de l'édition de code et de la gestion de projet classiques d'*IntelliJ*, il existe un certain nombre d'outils spécifiques au SDK *Android* qui sont accessibles directement depuis *Android Studio*.

L'émulateur

L'émulateur est compris dans le "pack" SDK de base. Il s'agit d'une machine *Android* virtuelle relativement complète et raisonnablement fonctionnelle, privée toutefois de certaines possibilités, parmi lesquelles :

- Pas d'accès au hardware du smartphone (USB, Bluetooth, NFC)
- Pas d'accès au GPS (il est toutefois possible d'imposer des coordonnées de localisation dans l'émulateur)
- Accès limité aux fonctions de téléphonie (bien sûr ni téléphone, ni SMS)
- Accès limité/contrôlé au réseau. L'émulateur est isolé de la machine et des autres émulateurs par un firewall virtuel. Lire à ce sujet : https://developer.android.com/studio/run/emulator-networking

L'exécution sur cible par USB

Pour les possesseurs d'un smartphone *Android*, il est possible et certainement préférable de faire exécuter le code directement sur la cible en connectant le smartphone à la machine de développement via un câble USB. L'installation d'un driver peut être requise dans certains rares cas ; se référer à la documentation de *Google* ou du constructeur de votre smartphone. Vous devrez activer le mode développeur sur le téléphone, puis autoriser le débogage par USB.

Editeur d'interfaces utilisateurs, éditeur XML

Android Studio contient un éditeur d'interfaces utilisateurs qui permet de générer des interfaces graphiques XML de manière *WYSIWYG*[1]. A peu près fonctionnel pour des interfaces relativement simples, cet éditeur montre vite ses limites lorsqu'il s'agit de composer des écrans plus sophistiqués. Il est généralement conseillé de faire tout de suite l'effort de comprendre la syntaxe de description d'interfaces XML d'*Android*, en sachant qu'on sera probablement tôt ou tard amené devoir le faire !

Logcat

L'outil probablement le plus précieux pour le débogage de vos applications : Logcat qui est la contraction de la commande catalog de Linux/Unix et de log file. La machine virtuelle ART (ou dalvik) envoie tous les événements concernant le mobile sur ce flot de sortie. Ce flot de sortie est donc aussi utilisable en production. Diverses librairies permettent de collecter les logs après le crash d'une application en production et de les envoyer au développeur qui pourra se livrer à un diagnostic plus précis. Par exemple : http://www.acra.ch/

Lors du développement, la sortie log est visible en temps réel, directement depuis *Android Studio*.

Log, System.out.println(), System.err.println()

Le complément à Logcat, qui va permettre de tracer l'exécution du code, également en production. Log est une classe permettant de générer des enregistrements de log qui vont apparaître avec un label (par exemple le nom de l'application) et un niveau de sévérité. Les println() sont généralement redirigés sur la même sortie que les Log, ce qui signifie qu'ils apparaîtront également dans le listing, mais sans identification particulière. Toutefois prenez l'habitude d'utiliser la classe Log plutôt que System.out/err, car non seulement cela vous permettra de filtrer les logs par leur sévérité, mais la classe Log permet en plus de passer une stacktrace d'exception en troisième paramètre, ce qui peut s'avérer très utile.

Google

La plus abondante source de renseignements pour le développement sur *Android* est sans conteste le net. En cas de problème, poser sa question (si possible en anglais, les forums francophones étant bien moins abondamment pourvus que les anglophones) directement sur la ligne de requête du moteur de recherche, éventuellement en la précédant du mot "android ». Par exemple, on peut imaginer une question sous la forme suivante :

- android popup dialog example

qui devrait vous donner des indications sur la manière de réaliser une fenêtre de dialogue. Le site sans conteste le plus souvent référencé, et le mieux pourvu en renseignements précieux est le bien connu www.stackoverflow.com (vérifiez que la question est similaire à la vôtre et cherchez en priorité la réponse marquée d'un vu vert). Mais les sites mis sur pied par *Google* pour les développeurs *Android*

[1] WYSIWYG : acronyme de "what you see is what you get"

sont aussi bien fournis en renseignements précieux : https://developer.android.com/docs/ et https://codelabs.developers.google.com/ L'important est de comprendre et d'être assez critique vis-à-vis du code que l'on trouve sur le net, bien souvent sa qualité laisse à désirer et il n'est plus forcément toujours d'actualité. En cas d'utilisation d'extrait de code du web, n'oublierez pas de citer la source ! Les plateformes mobiles évoluant très rapidement, vous apprendrez vite à jeter un œil à la date de la réponse ou du tutoriel.

De surcroît, même si votre collègue est meilleur codeur que vous, cela n'empêche pas d'essayer soi-même... C'est en effet le seul moyen pour qu'il ne reste pas définitivement meilleur que vous.