IFT1155 — Démonstration 1

Guillaume Poirier-Morency

Votre démonstrateur, en résumé:

- étudiant au baccalauréat en Informatique
- programmeur web (services web, systèmes distribués, base de données, async I/O, . . .)
- solide background en Python, Java, JavaScript et Vala
- père d'un petit garçon de 10 mois ;)
- disposé à répondre à vos questions et vous suggérer des approches pour résoudre vos problèmes

Tout le matériel de démonstration sera disponible depuis ma page du DIRO[1].

[1] http://www-etud.iro.umontreal.ca/~poirigui/ift1155-h16.html

Au menu...

- Android Studio
  - logcat
  - gradle
  - dépendances
  - VCS
- TP1
  - ZipFile
  - CSV
  - Modèle

Android Studio

Basé sur IntelliJ, un IDE développé par JetBrain.

- installation de l'environnement de développement
- familiarisation avec Android Studio et Gradle

Vote sur l'appareil

Il faut choisir deux appareils pour développer l'application Android:

- un téléphone
- une tablette

Vous pouvez consulter les différents appareils en ouvrant le *AVD Manager*.

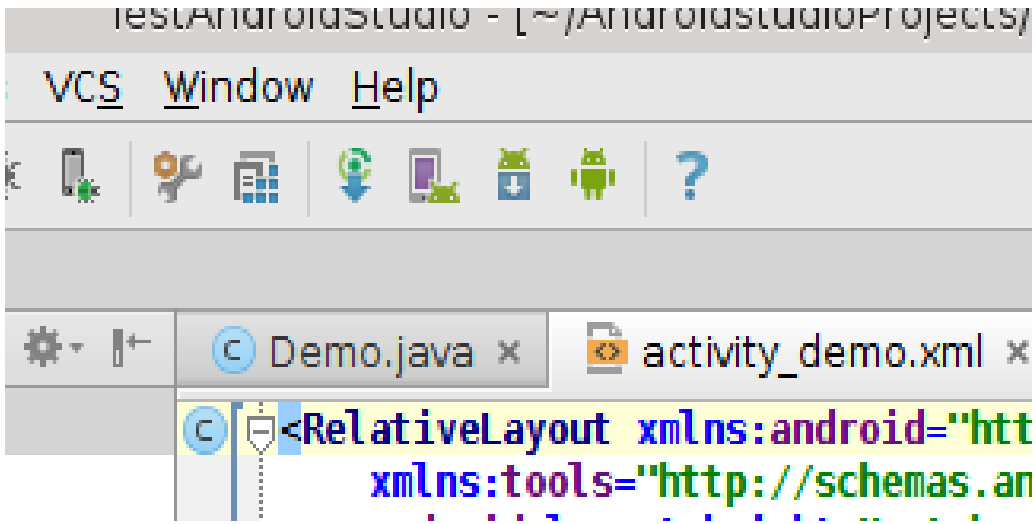

Figure 1: Utilitaires inclus dans Android Studio

logcat

Une fois le téléphone connecté à travers adb, on peut consulter les journaux à l'aide de l'outil logcat.

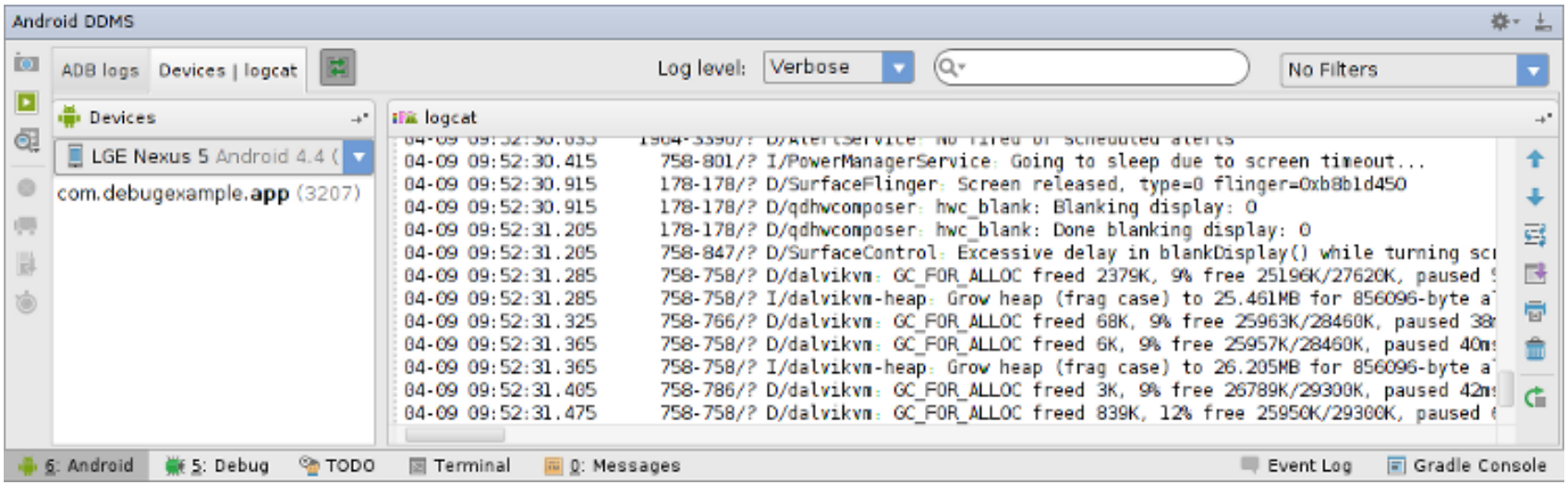

Figure 2: Aperçu des journaux systèmes (logcat)

Gradle

Gradle est un *build system* qui permet de construire des projets écrits en Java, Groovy et plusieurs autres langages de programmations.

Le système de *build* possède les caractéristiques suivantes:

- décrit à l'aide d'un DSL (Domain Specific Language) déclaratif
- modulaire
- gestion des dépendences
- extensible à l'aide de *plugins*
- permet de définir des *flavours* (ex. release, debug, . . .)

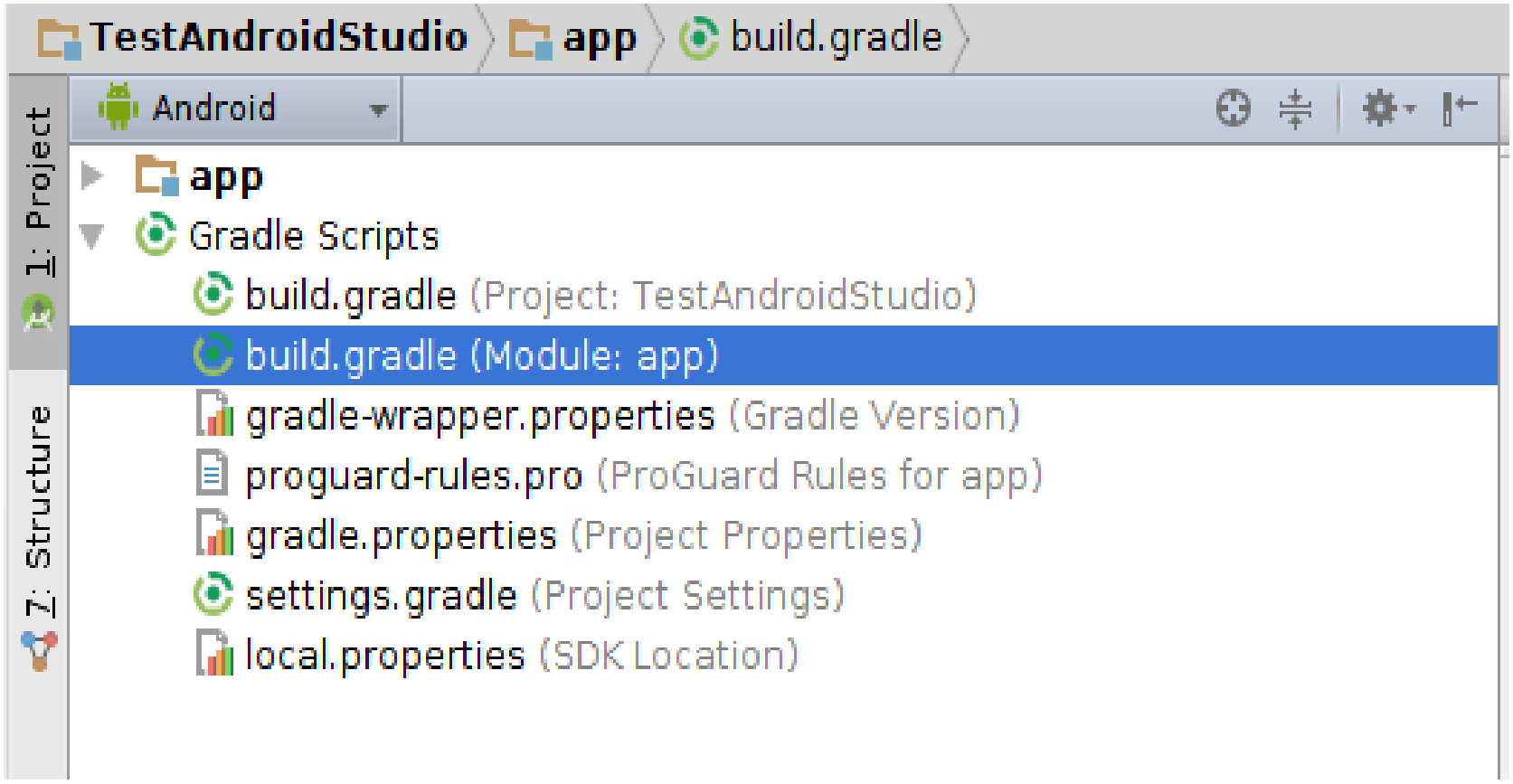

Figure 3: Accès aux scripts de Gradle

Dépendences

Les dépendences permettent d'inclure des librairies dans un projet.

Pour réaliser le travail pratique, nous aurons besoin d'un analyseur de CSV, Apache Commons CSV[2] vous est suggéré:

```
dependencies {
  compile group: 'org.apache.commons', name: 'commons-csv'
}
```

[2] https://commons.apache.org/proper/commons-csv/

VCS

Il est recommendé d'utiliser un système de contrôle de versions pour éviter des situations frustrantes.

L'intégration du VCS est fournie dans Android Studio. Je vous recommende fortement d'utiliser git.

Recommendations

- utiliser un modèle linéaire (ex. pas de branchements)
- créer un *commit* lorsque le projet fonctionne correctement et qu'une étape a été franchie (ex. bug résolu, fonctionnalité introduite, etc. . .)
- s'assurer que tous les fichiers sont suivis
- ne pas paniquer

Figure 4: Réalité du contrôle de versions

TP0

Ce TP vise à vous faire pratiquer la remise d'un travail avant qu'il ne soit trop tard.

L'énoncé est déjà disponible, vous n'avez qu'à suivre les instructions.

TP1

Le premier TP consiste à réaliser une application Android qui fournira à l'usager des itinéraires en transport en commun pour atteindre une destination de son choix.

Les points suivants seront abordés:

- ZipFile et CSV
- Données de la STM
- Modèle et algorithme

ZipFile

Les données fournies par la STM sont compressées sous format ZIP. Il faudra utiliser le paquet java.util.zip pour pouvoir manipuler le fichier.

- `ZipFile` l'archive
- `ZipEntry` une entrée de l'archive
- `java.io.InputStream` contenu d'une entrée de l'archive

```
ZipFile zipFile = new ZipFile (someFile);
ZipEntry zipEntry = zipFile.getEntry ("stops.txt");
InputStream entryStream =
  zipFile.getInputStream (zipEntry);
```

CSV

Les données conforme à la spécification GTFS sont encodés en CSV.

Apache Commons CSV sera utilisé pour interpréter l'objet de type `java.io.InputStream` qui est une *stream* de données encodées en CSV.

```
CSVParser parser =
  new CSVParser (new InputStreamReader (entryStream),
                 CSVFormat.RFC4180);

for (CSVRecord record : parser) {
  Log.d (record.get ("stop_id"));
}
```

Données de la STM (GTFS)

Table	Description
stops	Points de service
routes	Lignes d'autobus/métro
trips	Trajets d'autobus/métro
stop_times	Heures des arrêts

Les données devront être insérée dans une base de données SQLite pour pouvoir calculer efficacement les itinéraires.

Le schéma et les requêtes SQL complexes seront fournies avec l'énoncé.

Exemple de requête

Tous les arrêts à moins de 5 minutes de marche d'une coordonnée géographique donnée par le couple (?, ?):

```
select * from stops where
  51883 * (abs(stop_lat - ?) + abs(stop_lon - ?))
    <= 0.719 * 60;

select * from stops
  natural join stop_times
  where
    51883 * (abs(stop_lat - ?) + abs(stop_lon - ?))
      <= 0.719 * 60
  and
    departure_time between time ('now') and
                           time ('now', '+5 minutes');
```

Modèle (restreint)

On souhaite se rendre de A à B:

$$A \to A' \to B' \to B$$

Avec, au plus, une correspondance[3]:

$$A \to A' \to A'' \to B'' \to B' \to B$$

- A et B sont respectivement l'origine et la destination
- A', A'', B'' et B' sont des points de services de la STM
- temps au point A est noté $t(A)$
- distance entre A et B est noté $d(A, B)$

[3] les correspondances sont exponentiellement coûteuse à calculer dans ce modèle

Unités

- temps: secondes
- distance: m

Contraintes

- $d(A, A') + d(A'', B'') + d(B', B) < 1000$, au plus 1km de marche
- $t(A) < t(A') < t(A'') < t(B') < t(B)$, voyager dans le temps n'est pas permi

Calcul de distance

Longueur d'arc

La longueur d'un arc x résultant d'une différence d'angle Δ_θ est donné par:

$$x = r * rad(\Delta_\theta)$$

Distance de Manhattan

$$d(A, B) = |B_x - A_x| + |B_y - A_y|$$

Distance entre deux coordonnées

$$d(A,B) = r * rad(|B_{lat} - A_{lat}|) + rad(|B_{lon} - A_{lon}|) \quad (1)$$
$$= r * \frac{\pi}{180}(|B_{lat} - A_{lat}|) + \frac{\pi}{180}(|B_{lon} - A_{lon}|) \quad (2)$$
$$= \frac{\pi * r}{180} * (|B_{lat} - A_{lat}| + |B_{lon} - A_{lon}|) \quad (3)$$

Montréal forme un angle de $62.22°$[4] avec le méridien.

$$= \frac{\pi * r * \cos 62.22}{180} * (|B_{lat} - A_{lat}| + |B_{lon} - A_{lon}|) \quad (4)$$
$$\approx 51883.246273604 * (|B_{lat} - A_{lat}| + |B_{lon} - A_{lon}|) \quad (5)$$

Dans notre cas, $r = 6378.1km$ est donné par le rayon de la Terre[5].

[4] Déviation de la rue Pie-IX par rapport au méridien

[5] http://nssdc.gsfc.nasa.gov/planetary/factsheet/earthfact.html

Caclul de temps

Temps (à pieds)

En supposant une vitesse de 5km/h (1.39 m/s)

$$t_{marche}(A, B) = 0.719 * d(A, B)$$

Temps (STM)

Le temps est calculé selon un trajet sélectionné:

$$t(A, B) = t(B) - t(A)$$

Algorithme

1. Soit A, un point donné par (A_{lat}, A_{lon}) au temps $t(A)$
2. énumérer A' dans un rayons acceptable de marche (~1km) noté d_{marche} à partir de A
3. pour chaque $a \in A'$ énumérer B' dans un rayon résiduel de $1000 - d_{marche}$
4. énumérer les chemins entre A' et B' conformément au temps $t(A') = t(A) + 0.719 * d(A, A')$
5. énumérer les points intermédiaire A'', B'' entre A' et B' (difficile)
   - considérer les chemins entre A' et A''
   - considérer les chemins entre B'' et B'
   - considérer la distance de marche $d(A'', B'')$ résiduelle à $1000 - d(A, A') - d(B', B)$
6. sélectionner les k meilleurs chemins