



UV : LO52 – Rapport Travaux Pratiques

TP 1 : mise en place de l'environnement de
développement

1. Mise en place de l'environnement de développement

L'objectif est de faire une prise en main de l'outil Git, configurer et mettre en place l'environnement de développement.

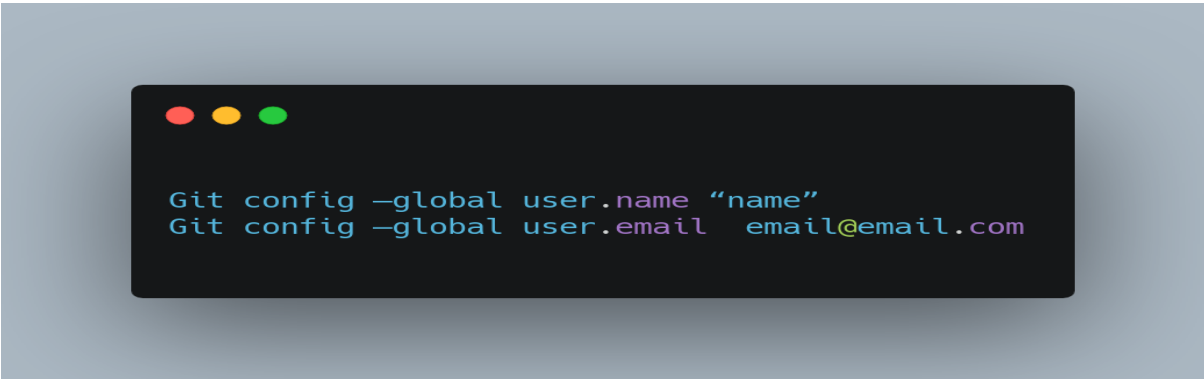
1.1. Installation et configuration de git

- Installation

Nous avons installé git sur notre machine en suivant les instructions, depuis le site de téléchargement de git : <https://git-scm.com/downloads>.

- Configuration

L'installation faite, nous avons procédé à la configuration de git avec nos informations d'identification sur github :



```
Git config --global user.name "name"  
Git config --global user.email email@email.com
```

Ensuite nous avons clone le dépôt git du projet sur github afin d'avoir une copie du code source sur notre machine.



```
Git clone https://github.com/gxfab/L052_A2020
```

Pour le suivi de notre travail, nous avons créé une branche git spécifique :



```
Git branch Agotsigedeon
```

Pour se positionner sur notre branche :



```
git checkout -b Agotsigedeon
```

Après avoir expliqué l'organisation de travail dans le fichier TP1.txt, on met à jour notre dépôt distant sur github.



```
Git add .  
Git commit -m "Mise à jour du fichier TP1.txt. "  
Git push -u origin Agotsigedeon.
```

Git étant configuré et l'accès au répertoire de travail étant fait, nous passons à l'étape de développement de l'application android.

1.2. Développement application android

L'objectif de cette partie est de pouvoir installer et prendre en main l'environnement de travail pour le développement android et ensuite développer une application "Hello world" de navigation entre différents activity.

- Installation

L'installation d'Android studio (IDE de développement android) est faite en suivant le guide officiel : <https://developer.android.com/studio/install>

- Réalisation du projet

Le projet consiste à créer deux activity android et ensuite passer de l'une à l'autre quand on clique sur un bouton. Nous développerons en JAVA.

Après avoir créé un "projet vide" avec l'assistant d'android studio, nous avons créé l'activity principale et son layout associé :

```
package com.example.helloworld;

import ...

public class MainActivity extends AppCompatActivity {

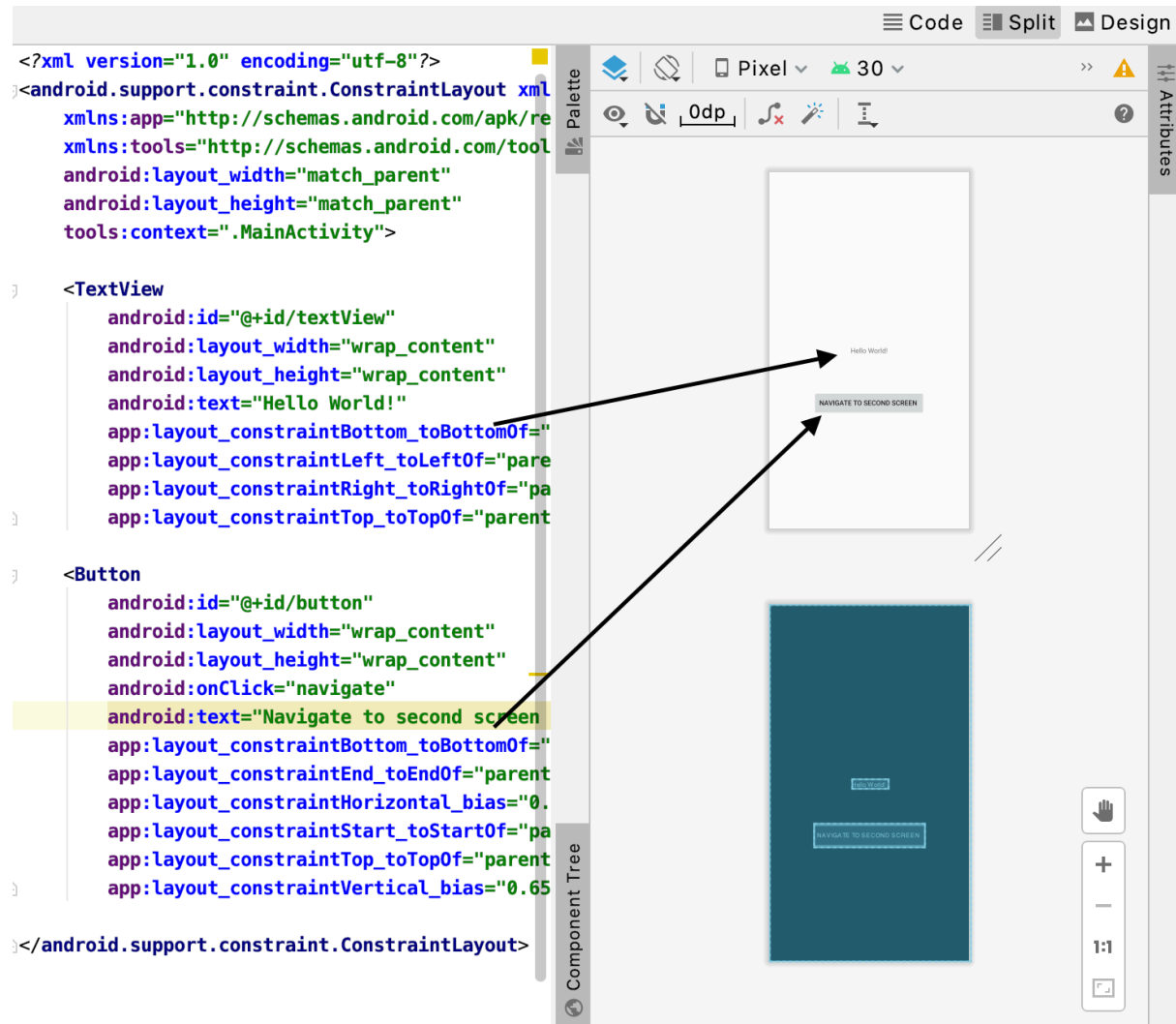
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    public void navigate(View view) {
        Intent intent = new Intent( packageContext: MainActivity.this, HelloActivity.class);
        startActivity(intent);
    }
}
```

Studio performance could be improved
Increasing the maximum heap size from...

Code source MainActivity

Il est associé à l'activity un fichier xml permettant de construire l'interface : on remarque la présence d'un texte « Hello Word » et d'un bouton intitulé : navigation to second screen.



Code source activity_main.xml

Ensuite nous, allons créer la seconde activity : `HelloActivity` , voici ci-dessous le code source de `HelloActivity.java` et son fichier xml `activity_hello.xml` décrivant l'interface

```
package com.example.helloworld;

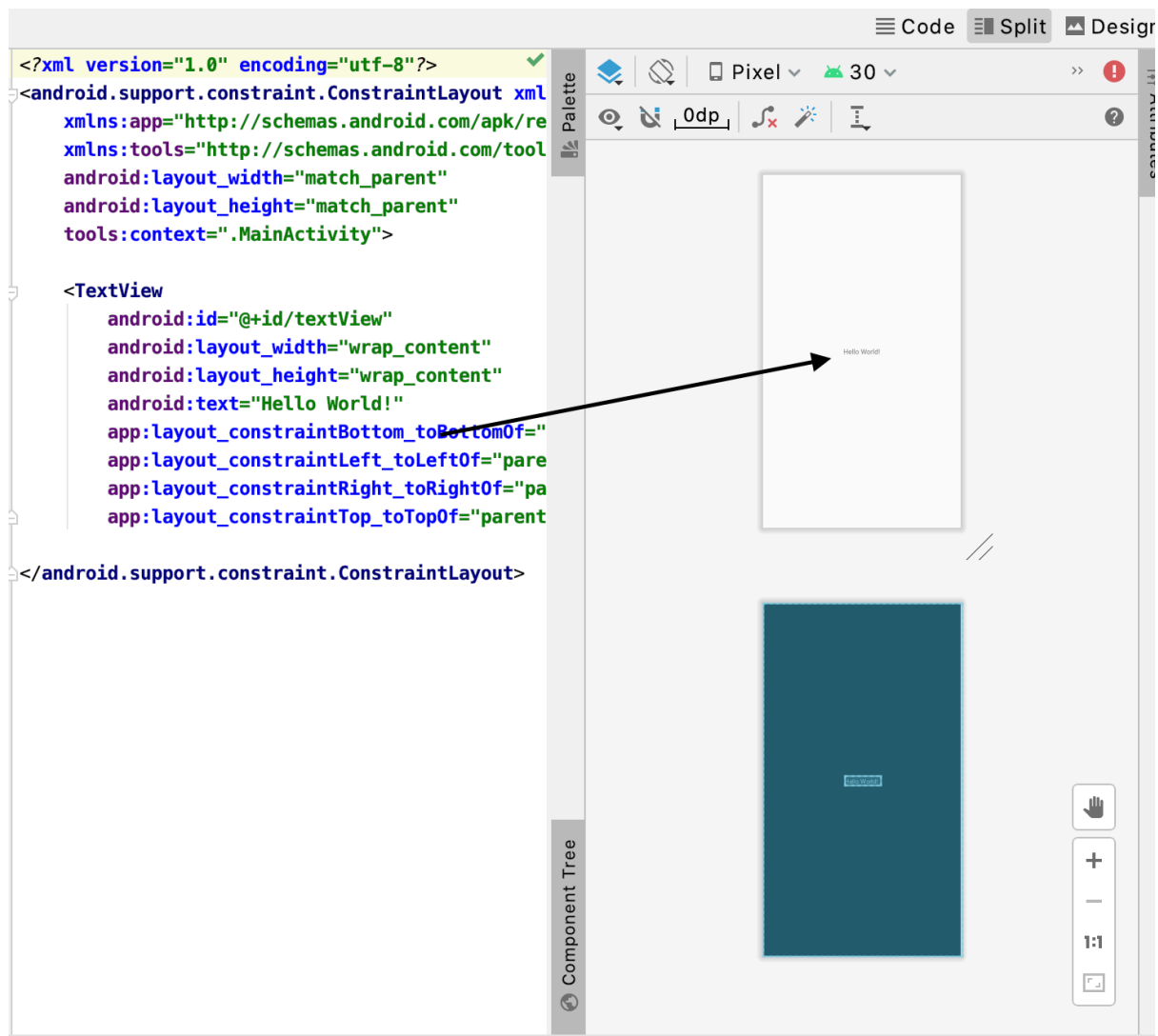
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;

public class HelloActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_hello);
    }
}
```

Code source HelloActivity

N avons déposé le composant textView dans le layout lié à HelloActivity affichant : Hello Word



Code source activity_hello.xml

Le fichier string.xml contient la valeur des textes affichés sur nos composants, nous permettant de la modification de nos intitulés :

```
<resources>
<string name="app_name">Main Activity</string>
<string name="hello_activity_name">Hello World Activity</string>
</resources>
```

Pour pouvoir passer du Main Activity vers le Hello Activity, nous avons créé la méthode navigation qui lors d'un événement `onClick` du bouton spécifié dans `l'activity_main.xml`


```

<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:onClick="navigate"
    android:text="Navigate to second screen "
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.498"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.657" />

```

```

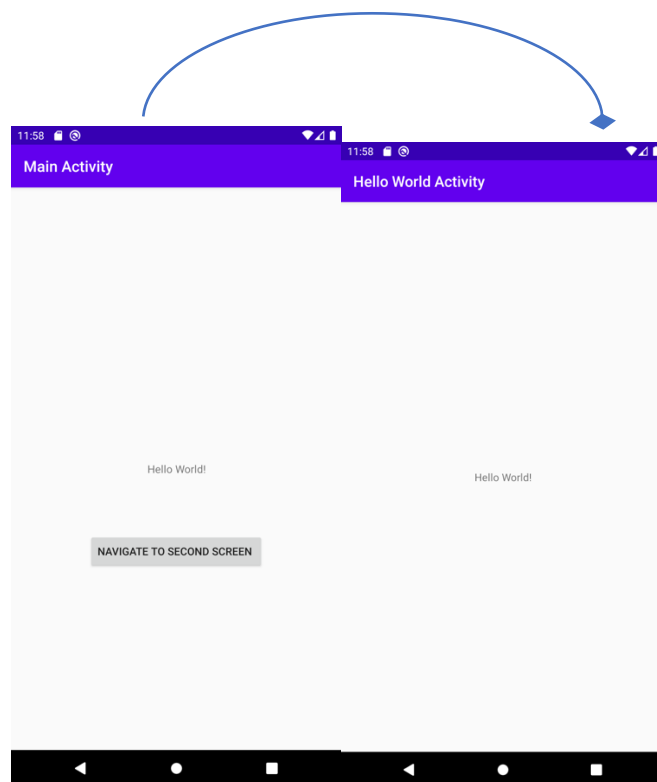
public void navigate(View view) {
    Intent intent = new Intent(packageContext: MainActivity.this, HelloActivity.class);
    startActivity(intent);
}

```

Code source méthode navigate et bouton de navigation

1.3. Démo

Nous avons pu obtenir le résultat suivant :



Demo TP1