

Android - wprowadzenie

Kraków, 14.11.2017

Magdalena Ludwin
Ewa Brzeziecka
Emilia Lubos

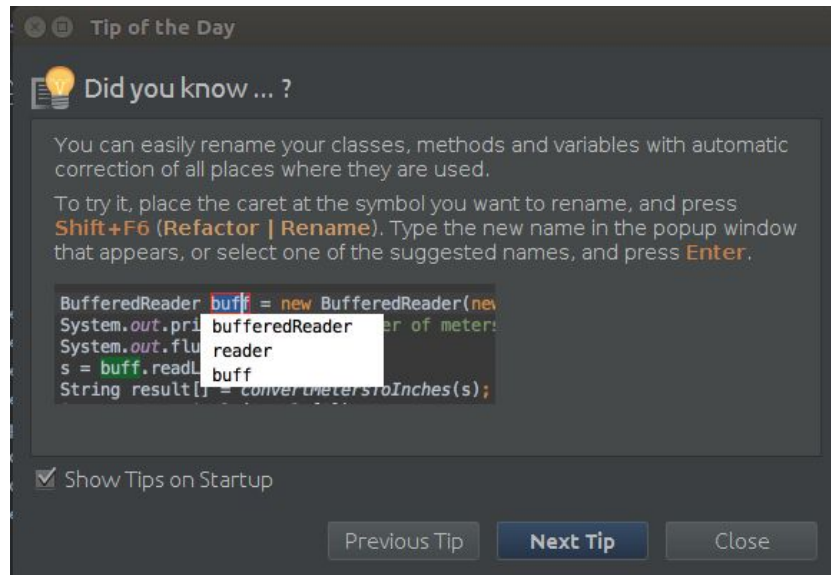
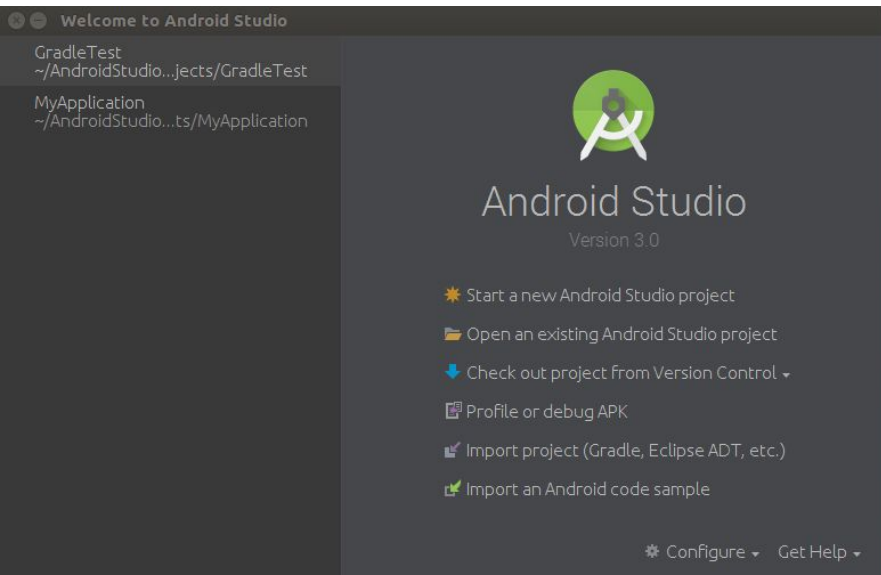


Android Studio



Android
Studio

IDE udostępnione bezpłatnie przez Google dla developerów systemu Android. Bazuje na IntelliJ IDEA.



Project

GradleTest ~/AndroidStudioProjects/GradleTest

- .gradle
- .idea
- app
 - build
 - libs
 - src
 - .gitignore
 - app.iml
 - build.gradle
 - proguard-rules.pro
- build
- gradle
 - wrapper
 - gradle-wrapper.jar
 - gradle-wrapper.properties
 - .gitignore
 - build.gradle
 - gradle.properties
 - GradleTest.iml

Build Variants

Module	Build Variant
app	debug

Terminal

```

+ hellokitty@hellokitty-Latitude-E6540:~/AndroidStudioProjects/GradleTest$

```

activity_main.xml x MainActivity.java x app x gradle.properties x gradlew x

android{} buildTypes{}

```

1  apply plugin: 'com.android.application'
2
3  android {
4      compileSdkVersion 26
5      defaultConfig {
6          applicationId "com.example.hellokitty.gradletest"
7          minSdkVersion 15
8          targetSdkVersion 26
9          versionCode 1
10         versionName "1.0"
11         testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
12     }
13     buildTypes {
14         release {
15             minifyEnabled false
16             proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
17         }
18     }
19 }
20
21 dependencies {
22     implementation fileTree(include: ['*.jar'], dir: 'libs')
23     implementation 'com.android.support:appcompat-v7:26.1.0'
24     implementation 'com.android.support.constraint:constraint-layout:1.0.2'
25     testImplementation 'junit:junit:4.12'

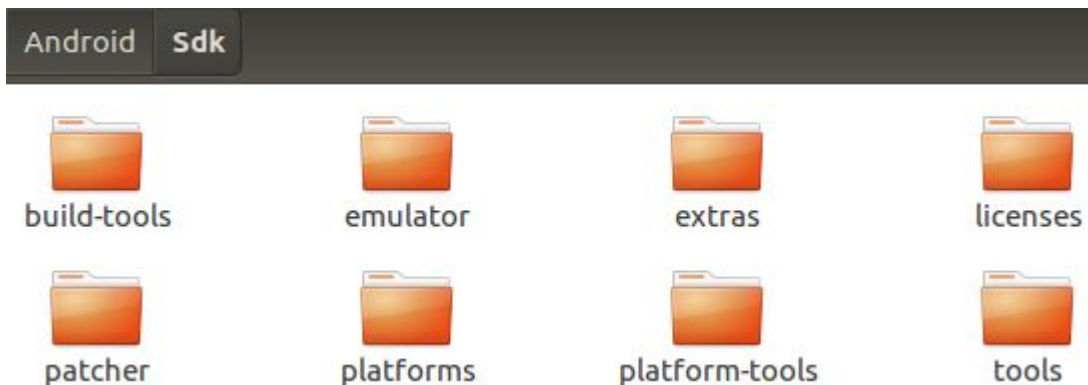
```

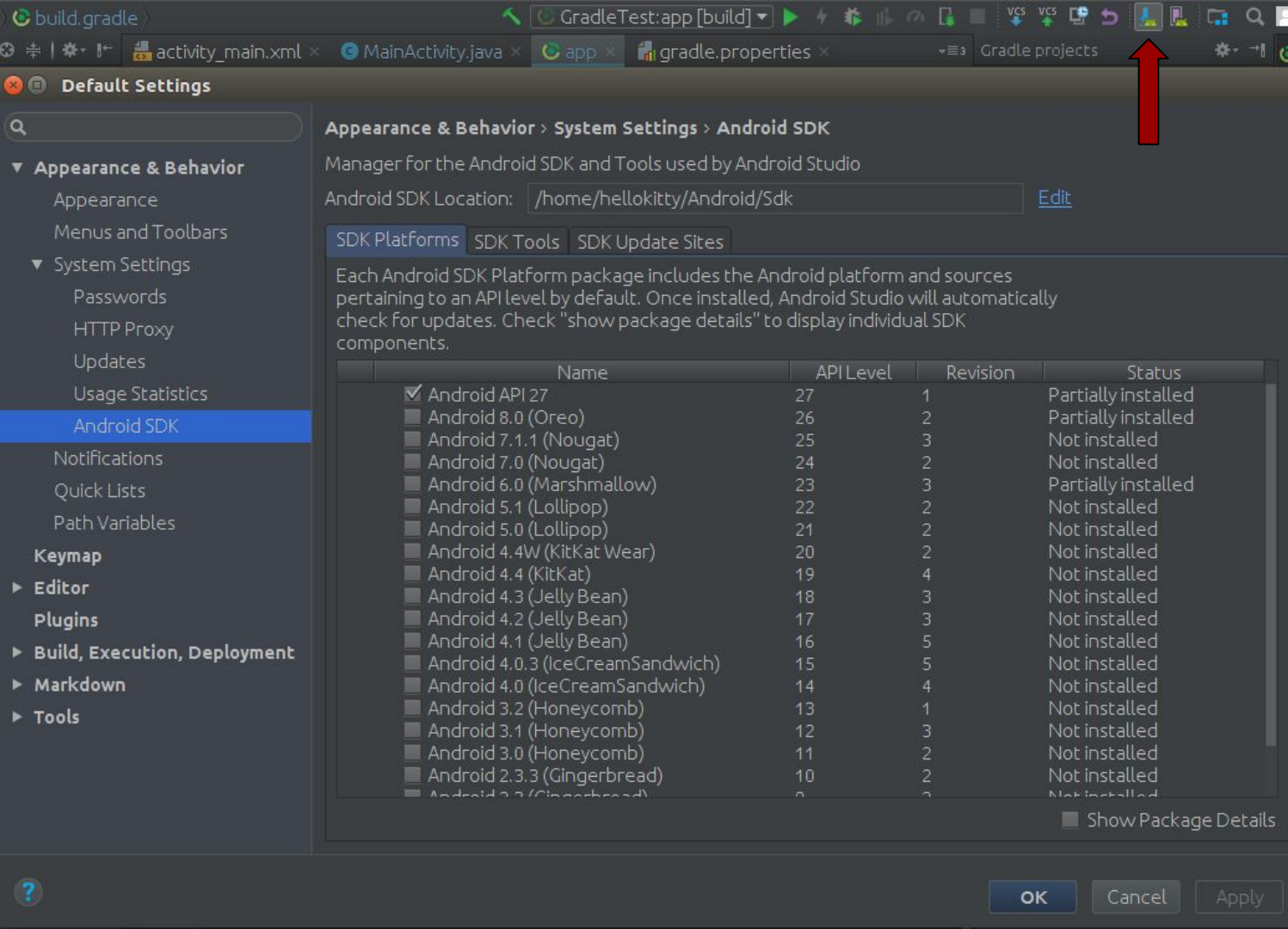
Gradle projects

- GradleTest
 - GradleTest (root)
 - Tasks
 - Run Configurations
 - :app
 - Tasks
 - android
 - build
 - assemble
 - assembleAndroidTest
 - assembleDebug
 - assembleRelease
 - build
 - buildDependents
 - buildNeeded
 - clean
 - cleanBuildCache
 - compileDebugAndroidTest
 - compileDebugSources
 - compileDebugUnitTest
 - compileReleaseSources
 - compileReleaseUnitTest
 - mockableAndroidJar

Android SDK

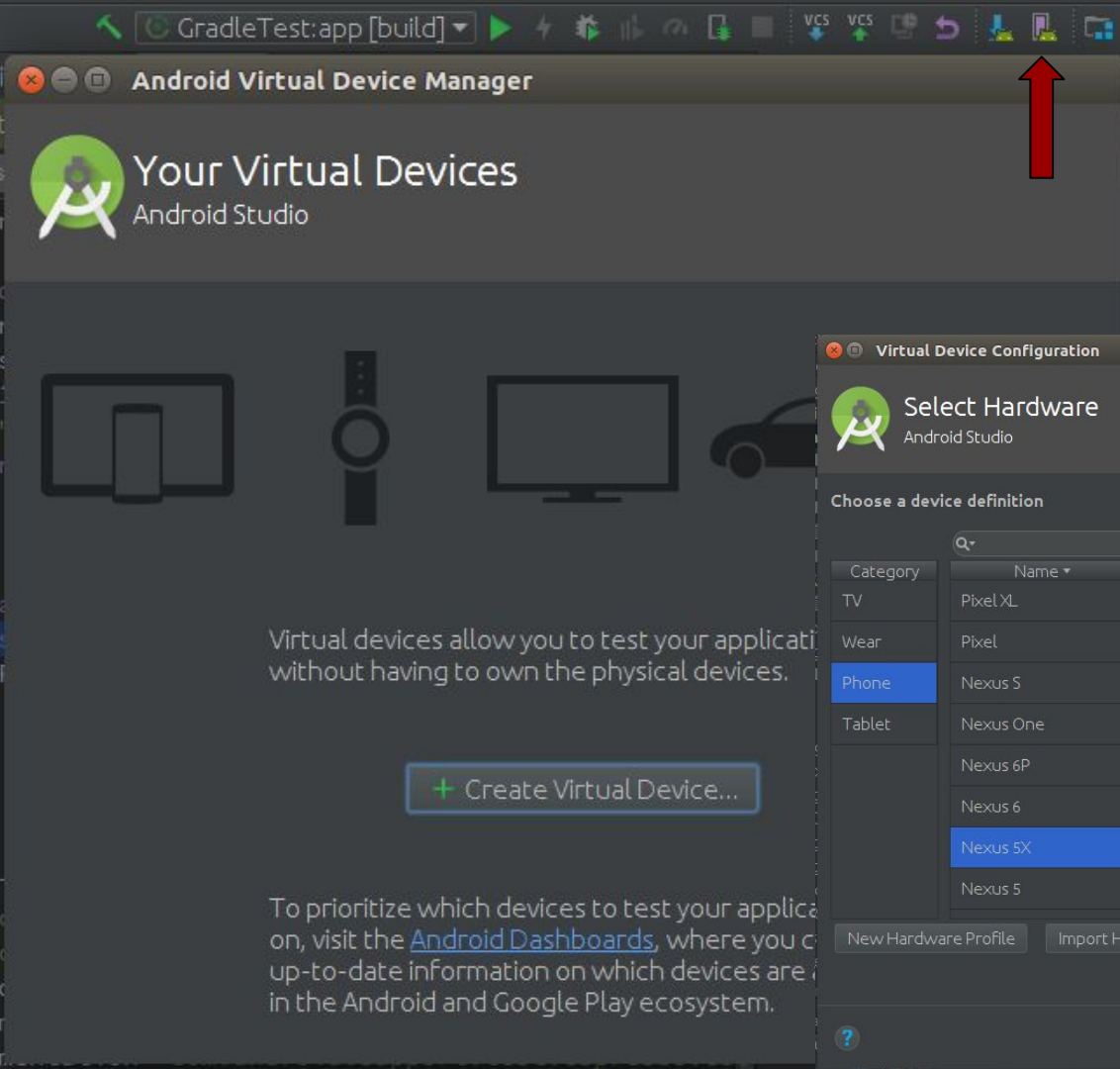
Zestaw narzędzi dla programistów przeznaczony do tworzenia aplikacji na platformę Android. Zawiera przykładowe projekty, tutoriale, biblioteki, emulator, debugger oraz inne narzędzia. Aplikacje są pisane w języku Java.





SDK Manager

Pozwala
doinstalować
brakujące
moduły



ADV Manager

Gradle

Narzędzie do automatyzacji procesu budowania projektu. Innymi narzędziami są np. Ant i Maven. Gradle korzysta z języka Groovy.

Reguła „convention over configuration”.
Zminimalizowanie potrzebnej konfiguracji,
poprzez używanie gotowych wartości
domyślnych.

Pierwszy skrypt budujący

Plik `build.gradle` powstaje automatycznie przy tworzeniu nowego projektu w Android Studio. Utworzymy nowy projekt z domyślną konfiguracją.



Create Android Project

Application name

GradleTest

Company domain

hellokitty.example.com

Project location

/home/hellokitty/AndroidStudioProjects/GradleTest

Package name

com.example.hellokitty.gradletest

Edit

- ☐ Include C++ support
- ☐ Include Kotlin support

Previous

Next

Cancel

Finish

Create New Project



Target Android Devices

Select the form factors and minimum SDK

Some devices require additional SDKs. Low API levels target more devices, but offer fewer API features.

☒ Phone and Tablet

API 15: Android 4.0.3 (IceCreamSandwich)

By targeting **API 15 and later**, your app will run on approximately **100%** of devices.

- ☐ Include Android Instant App support

☐ Wear

API 21: Android 5.0 (Lollipop)

☐ TV

API 21: Android 5.0 (Lollipop)

☐ Android Auto☐ Android Things

API 24: Android 7.0 (Nougat)

Previous

Next

Cancel

Finish



Add an Activity to Mobile



Add No Activity



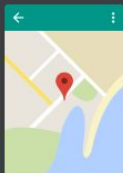
Basic Activity



Bottom Navigation Activity



Empty Activity



Previous

Next

Cancel

F

Create New Project



Configure Activity



Creates a new empty activity

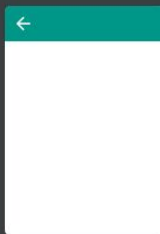
Activity Name

MainActivity

☒ Generate Layout File

Layout Name

activity_main

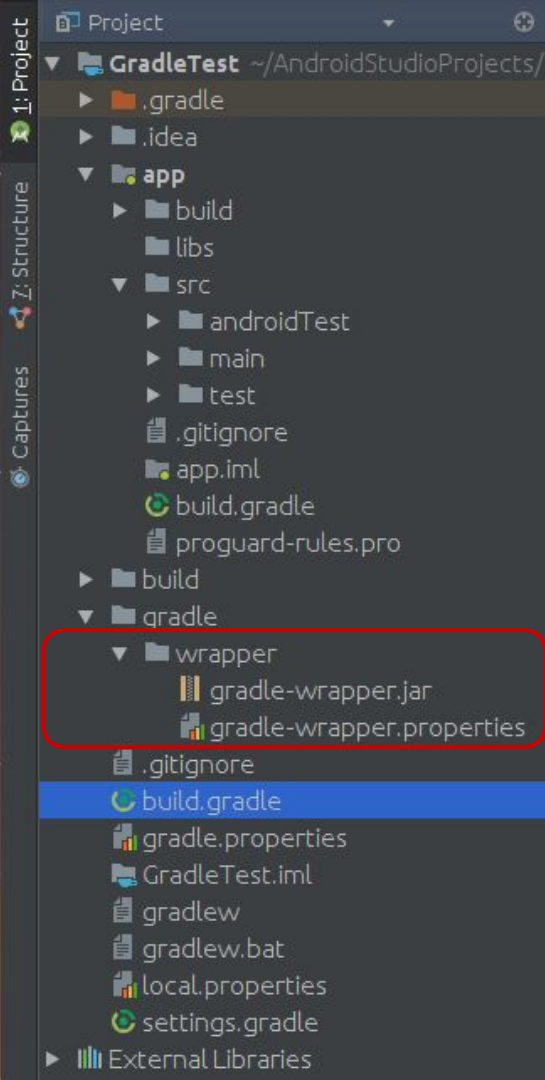
☒ Backwards Compatibility (AppCompat)

Previous

Next

Cancel

Finish



Struktura projektu

Po utworzeniu projektu, możemy zobaczyć jego strukturę w panelu po lewej. Gdy wejdziemy w GradleTest->app znajdziemy plik build.gradle. Jest to skrypt w języku Groovy. Opisuje proces budowania projektu. Niżej, w katalogu gradle znajduje się wrapper. Dzięki temu nie musimy instalować Gradle na naszym komputerze. Drugi plik app/build.gradle dotyczy konkretnego modułu.

```
// Top-level build file where you can add configuration options
// common to all sub-projects/modules.

buildscript {

    repositories {
        google()
        jcenter()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:3.0.0'

        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}

allprojects {
    repositories {
        google()
        jcenter()
    }
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

Sekcja buildscript służy do konfigurowania samego skryptu. Ustawia repozytoria Google i JCenter do pobierania zależności wymaganych przez skrypt. Następnie zdefiniowana jest wersja Gradle, z której korzystamy. Blok allprojects konfiguruje repozytoria i dependencje definiowane we wszystkich modułach projektu.

```

1  apply plugin: 'com.android.application'
2
3  android {
4      compileSdkVersion 26
5      defaultConfig {
6          applicationId "com.example.hellokitty.gradletest"
7          minSdkVersion 15
8          targetSdkVersion 26
9          versionCode 1
10         versionName "1.0"
11         testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
12     }
13     buildTypes {
14         release {
15             minifyEnabled false
16             proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
17         }
18     }
19 }
20
21 dependencies {
22     implementation fileTree(dir: 'libs', include: ['*.jar'])
23     implementation 'com.android.support:appcompat-v7:26.1.0'
24     implementation 'com.android.support.constraint:constraint-layout:1.0.2'
25     testImplementation 'junit:junit:4.12'
26     androidTestImplementation 'com.android.support.test:runner:1.0.1'
27     androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.1'
28 }
29

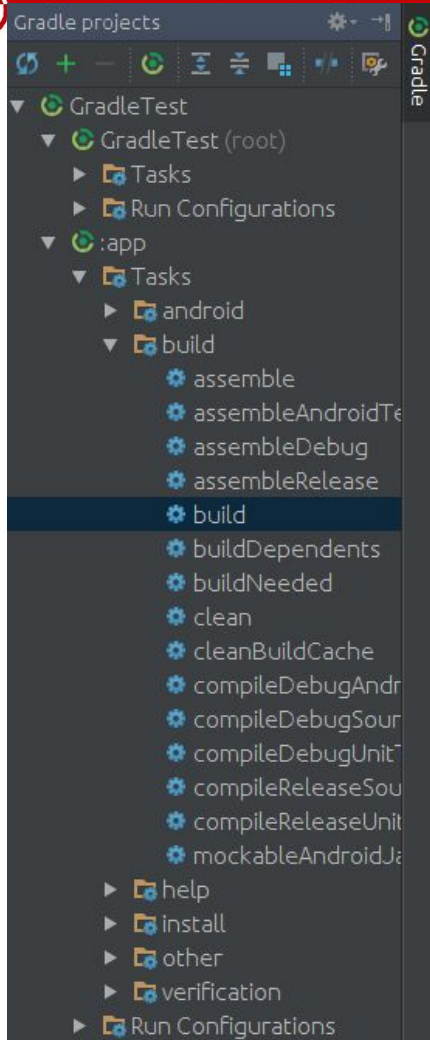
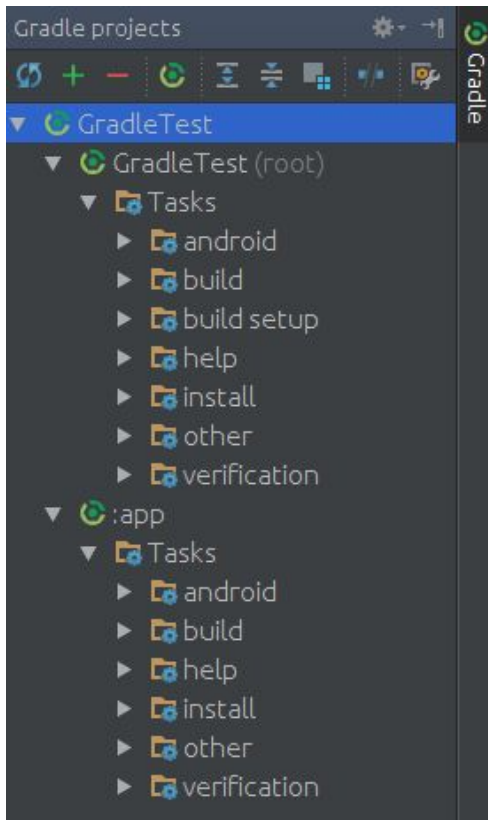
```

app/build.gradle

Są tutaj elementy
specyficzne dla
platformy Android:

docelowa i minimalna
wersja Android SDK,
wersja narzędzia
budującego oraz SDK
wykorzystywane do
kompilowania. Skrypt
nadpisuje informacje z
AndroidManifest.xml

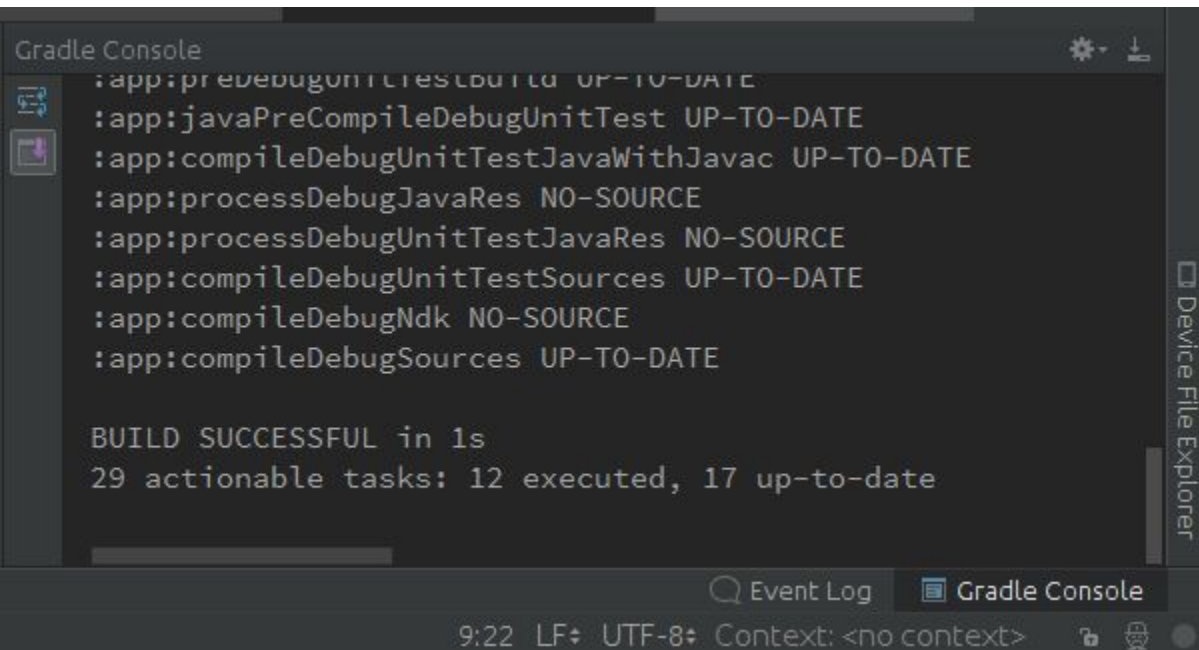
Blok buildTypes opisuje możliwe buildy. W przykładzie release, jednak domyślnie tworzone są i tak release i debug. Sekcja dependencies definiuje zależności projektu. Dodaje pliki JAR do folderu libs.



Taski Gradle

Po prawej stronie mamy dostępne taski Gradle. Wybrany task uruchamiamy podwójnym kliknięciem.

Konsola Gradle



```
Gradle Console
:app:preDebugUnitTestBuild UP-TO-DATE
:app:javaPreCompileDebugUnitTest UP-TO-DATE
:app:compileDebugUnitTestJavaWithJavac UP-TO-DATE
:app:processDebugJavaRes NO-SOURCE
:app:processDebugUnitTestJavaRes NO-SOURCE
:app:compileDebugUnitTestSources UP-TO-DATE
:app:compileDebugNdk NO-SOURCE
:app:compileDebugSources UP-TO-DATE

BUILD SUCCESSFUL in 1s
29 actionable tasks: 12 executed, 17 up-to-date

Device File Explorer

Event Log Gradle Console
9:22 LF+ UTF-8+ Context: <no context>
```

W prawym dolnym rogu znajduje się konsola Gradle. Możemy zobaczyć w niej wynik wykonania tasków. Z głównego menu wybrać: Build -> Make Project

Przydatne w przypadku szukania błędów.



Terminal

```
Terminal
+ hellokitty@hellokitty-Latitude-E6540:~/AndroidStudioProjects/GradleTest$ ./gradlew tasks
x > Task :tasks

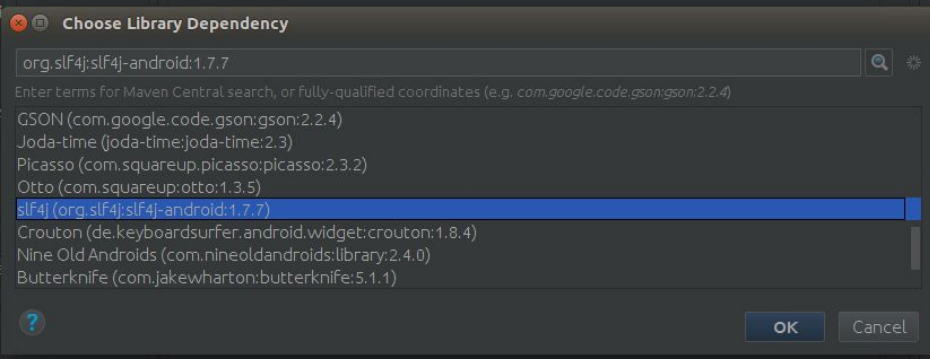
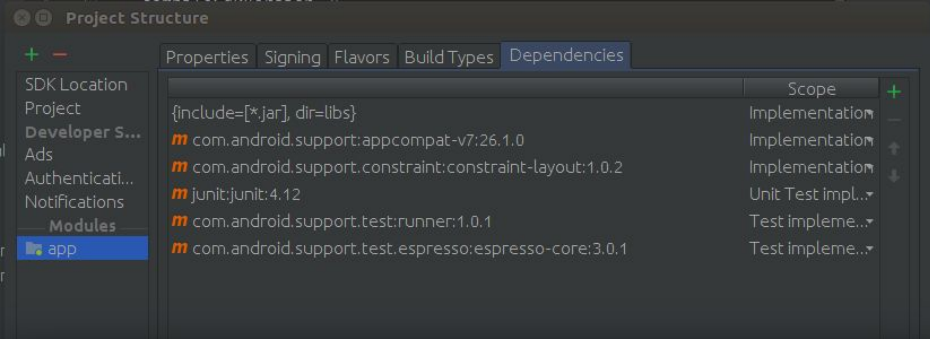
-----
All tasks runnable from root project
-----

Android tasks
-----
androidDependencies - Displays the Android dependencies of the project.
signingReport - Displays the signing info for each variant.
sourceSets - Prints out all the source sets defined in this project.

Build tasks
-----
assemble - Assembles all variants of all applications and secondary packages.
assembleAndroidTest - Assembles all the Test applications.
assembleDebug - Assembles all Debug builds.
```

`./gradlew tasks`
komenda
wypisze
dostępne taski.

`./gradlew <task>`
wykonanie taska.



Dodawanie dependencji

1. Bezpośrednio przez plik build.gradle, lub:
2. PPM app -> Open Module Settings

```
dependencies {  
    implementation fileTree(include: ['*.jar'], dir: 'libs')  
    implementation 'com.android.support:appcompat-v7:26.1.0'  
    implementation 'com.android.support.constraint:constraint-layout:1.0.2'  
    testImplementation 'junit:junit:4.12'  
    androidTestImplementation 'com.android.support.test:runner:1.0.1'  
    androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.1'  
    implementation 'org.slf4j:slf4j-android:1.7.7'  
}
```

Resource Shrinking

```
android {  
    compileSdkVersion 26  
    defaultConfig {  
        applicationId "com.example.hellokitty.gradletest"  
        minSdkVersion 15  
        targetSdkVersion 26  
        versionCode 1  
        versionName "1.0"  
        testInstrumentationRunner "android.support.test.runner.AndroidJUnit4"  
    }  
    buildTypes {  
        release {  
            minifyEnabled true  
            shrinkResources true  
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  
        }  
    }  
}
```

Aby automatycznie
usunąć nieużywane
źródła w czasie
budowania należy
dodać do pliku
app/build.gradle

shrinkResources true

w sekcji release



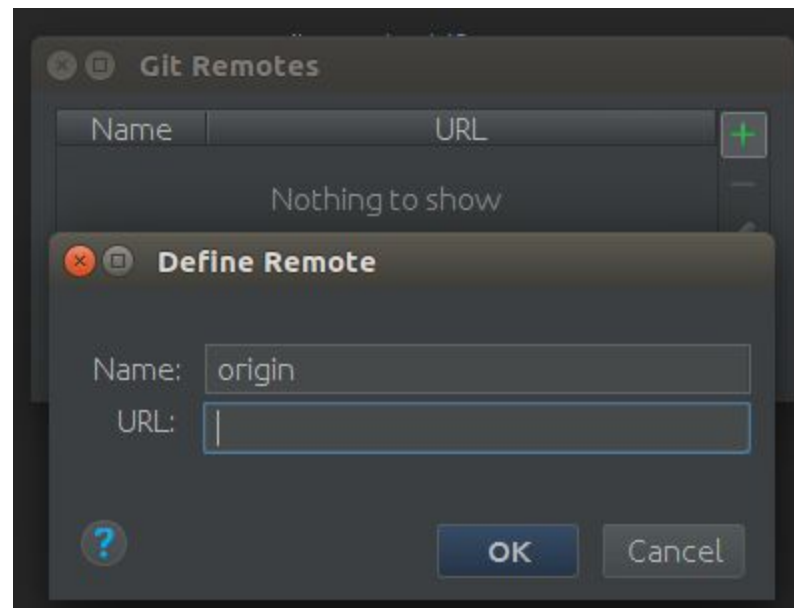
Version Control System - Git

Musimy mieć zainstalowanego Gita w systemie. Wtedy: VCS -> Enable Version Control Integration

VCS -> Git -> Remotes

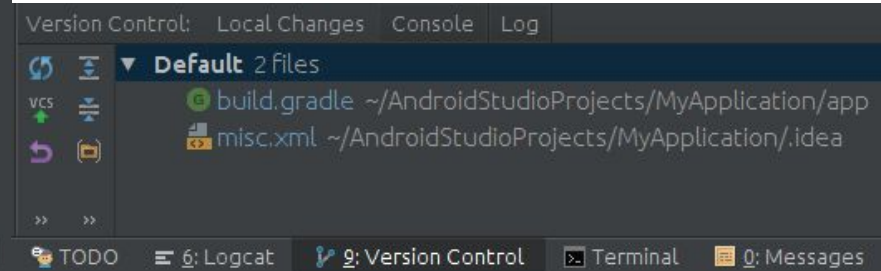
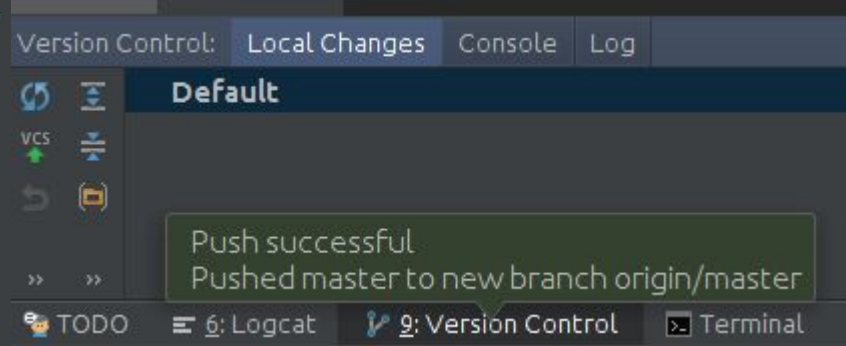
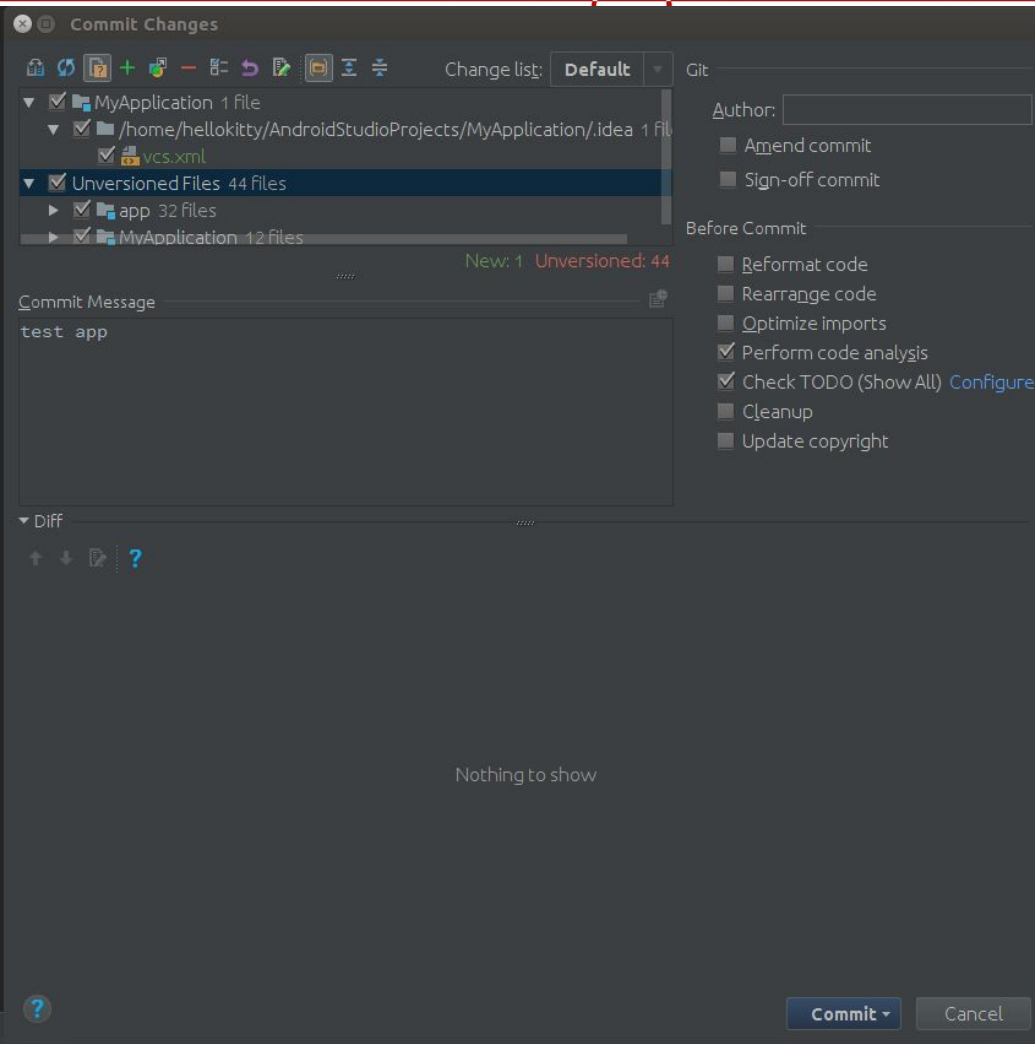
Define Remote

URL: <link do naszego repo>



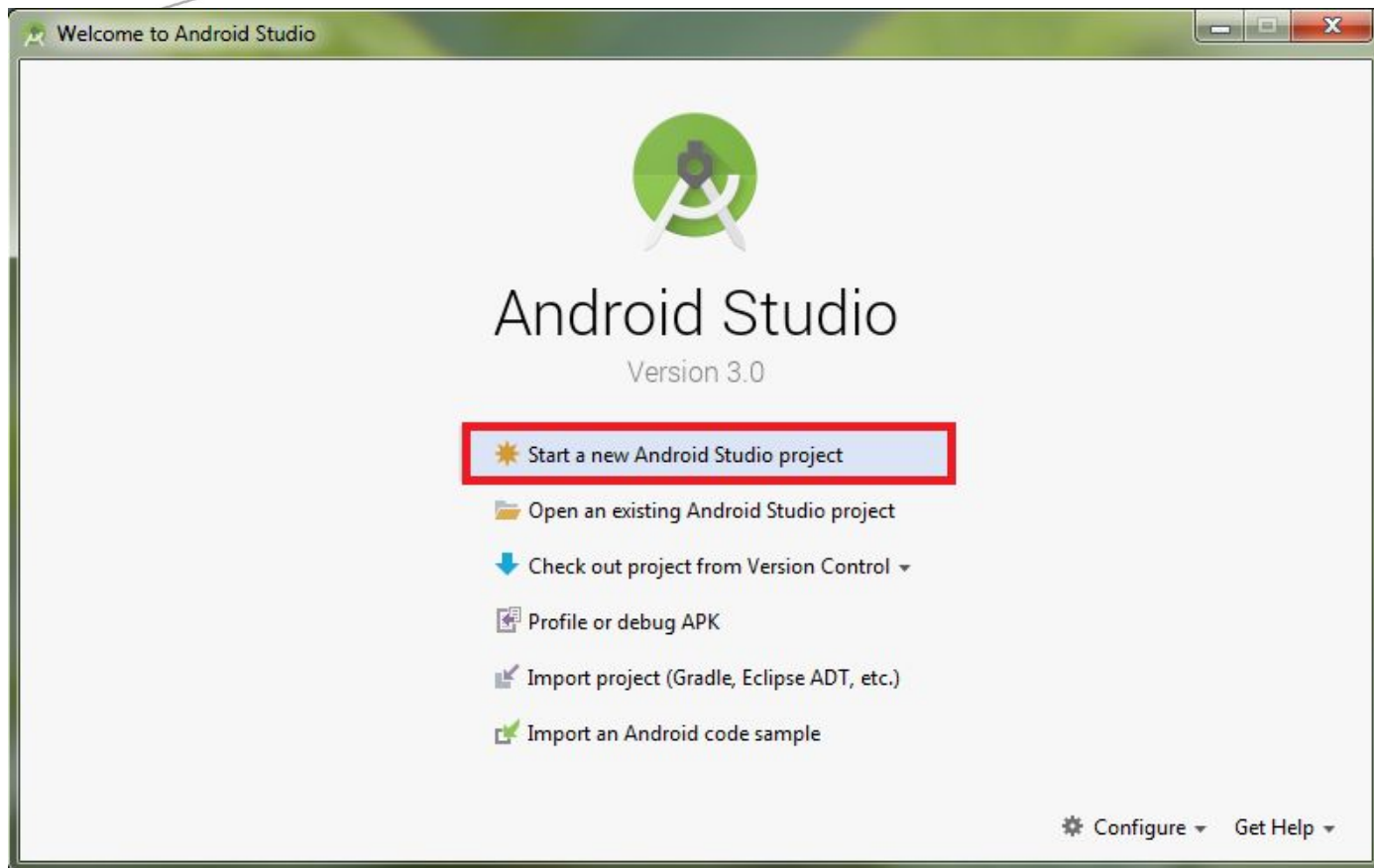
VCS -> Commit changes

Commit and Push



Hello World!

Pierwsza aplikacja



Create New Project

Create Android Project

Application name
HelloWorld

Company domain
emilia.agh.pl

Project location
C:\Users\Emilia\AndroidStudioProjects\HelloWorld

Package name
pl.agh.emilia.helloworld Edit

☐ Include C++ support

☐ Include Kotlin support

Previous **Next** Cancel Finish

Create New Project

Target Android Devices

Select the form factors and minimum SDK

Some devices require additional SDKs. Low API levels target more devices, but offer fewer API features.

☒ **Phone and Tablet**
API 21: Android 5.0 (Lollipop)
By targeting **API 21 and later**, your app will run on approximately **71,3%** of devices. [Help me choose](#)
☐ Include Android Instant App support

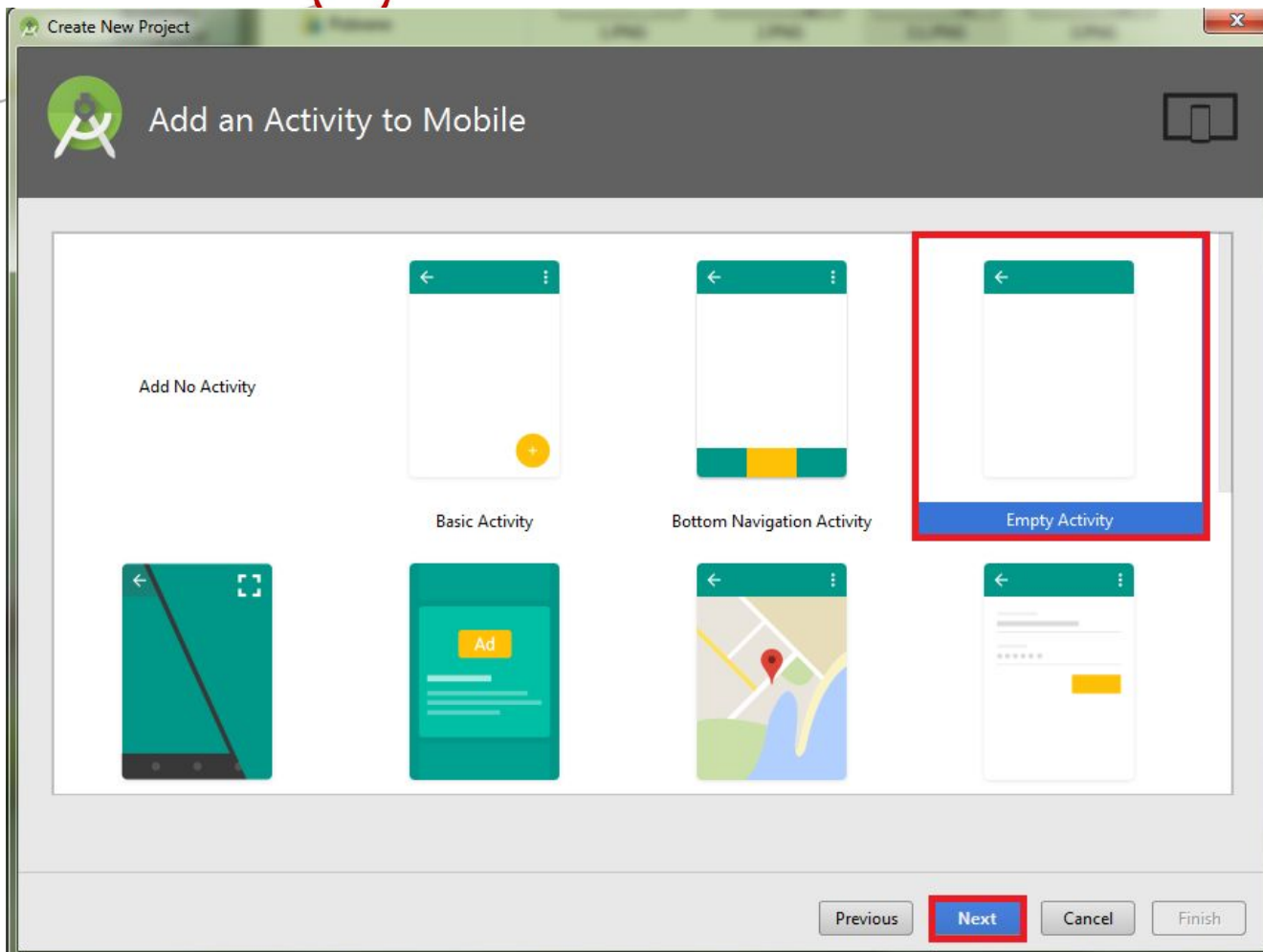
☐ **Wear**
API 21: Android 5.0 (Lollipop)

☐ **TV**
API 21: Android 5.0 (Lollipop)

☐ **Android Auto**

☐ **Android Things**
API 24: Android 7.0 (Nougat)

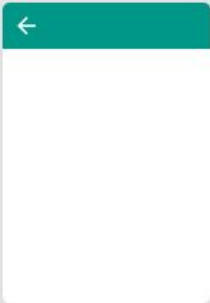
Previous **Next** Cancel Finish



Create New Project

Configure Activity

Creates a new empty activity



Activity Name

MainActivity

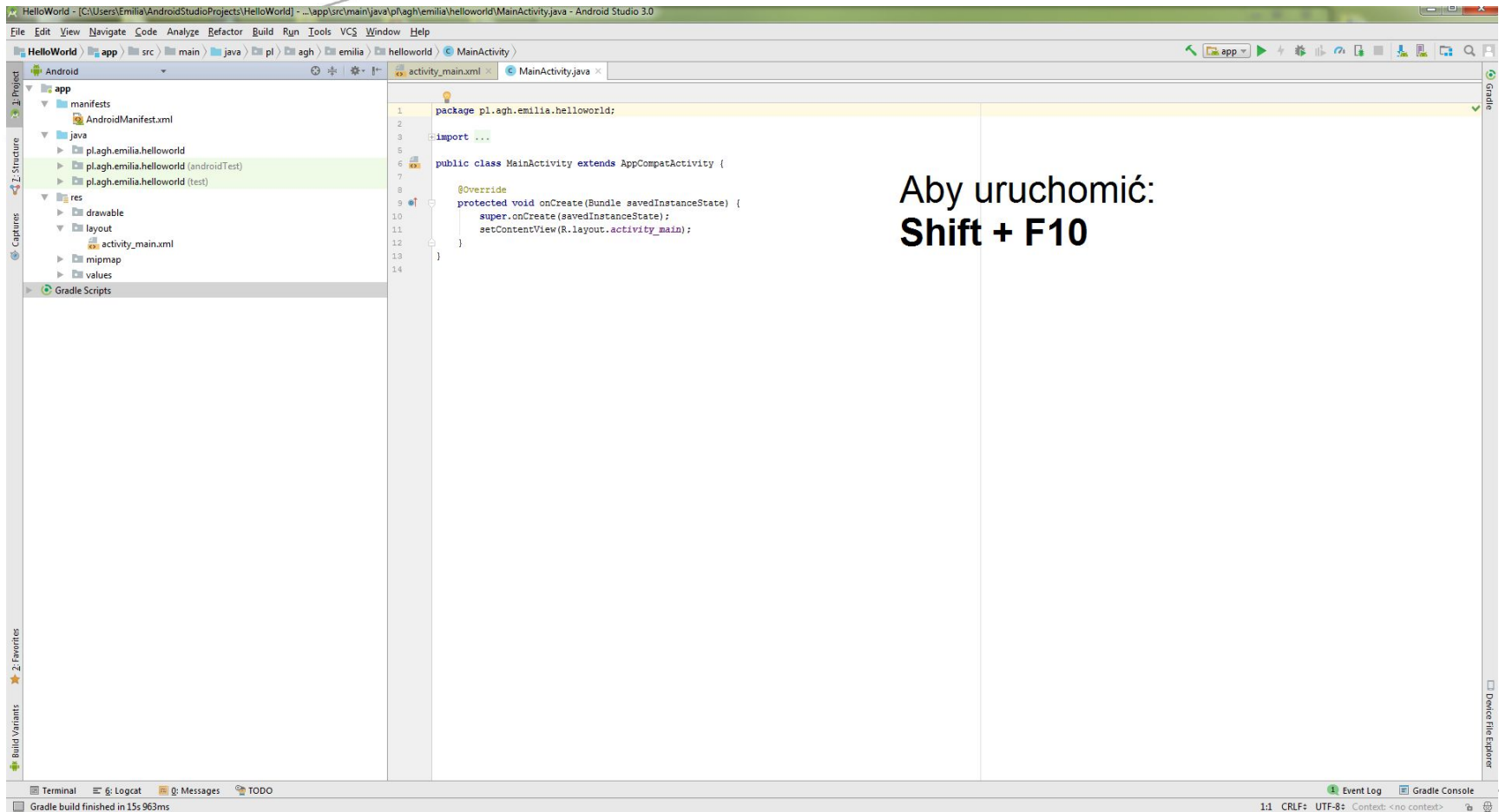
☒ Generate Layout File

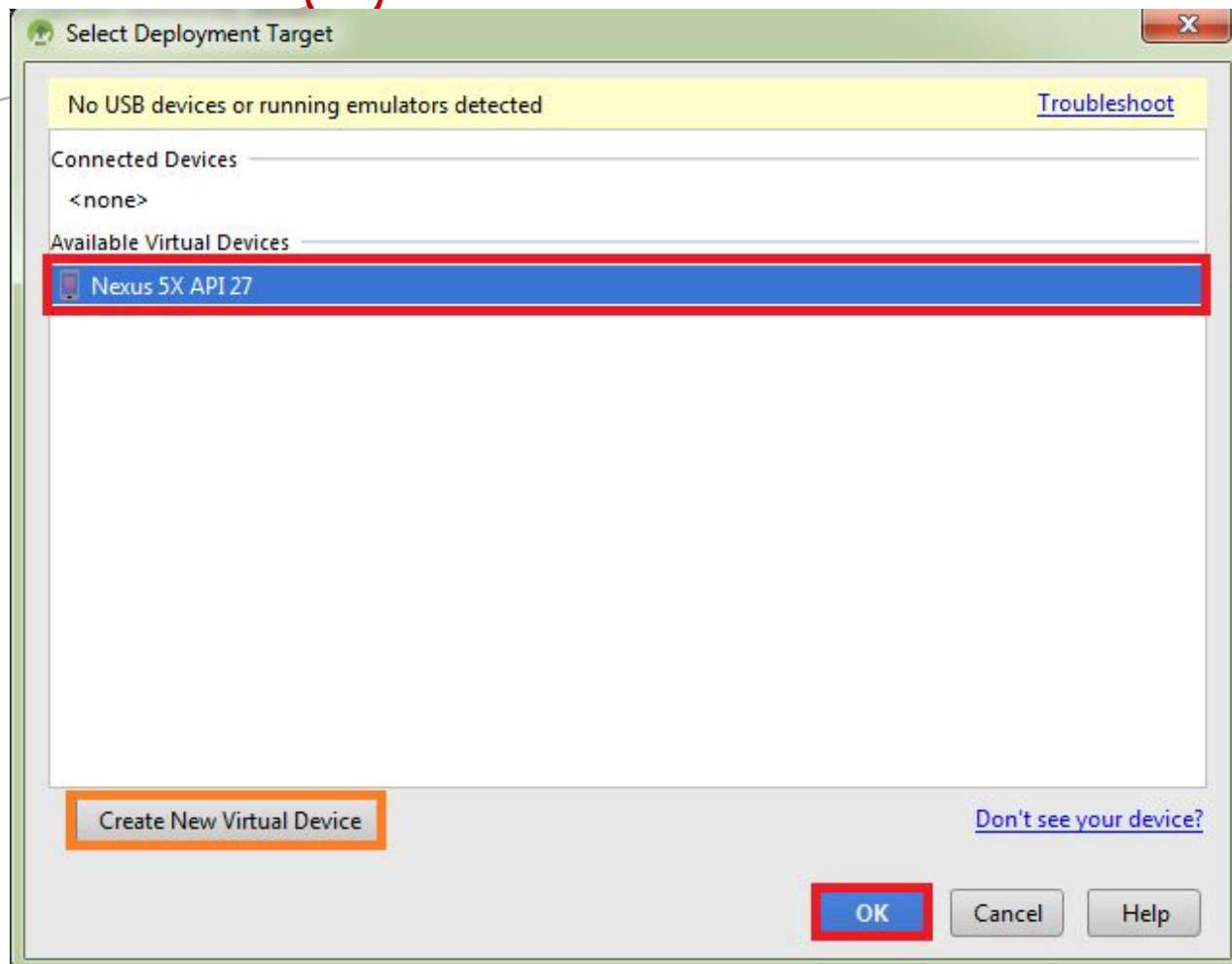
Layout Name

activity_main

☒ Backwards Compatibility (AppCompat)

Previous Next Cancel **Finish**







Drzewo projektu

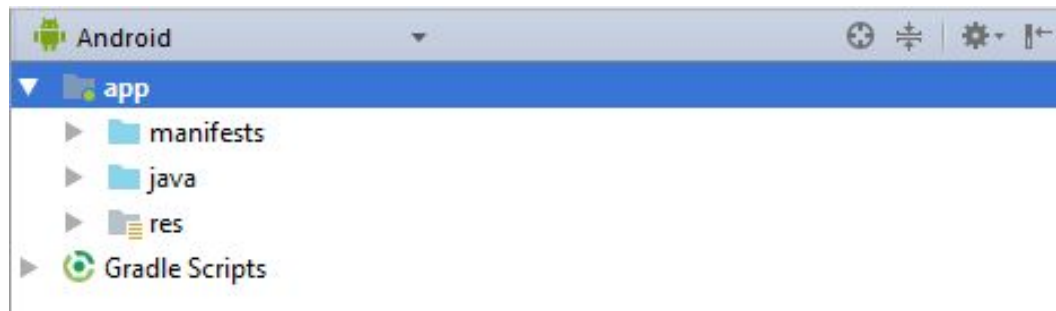


Drzewo projektu

Znajdują się tu pliki i zasoby potrzebne do zbudowania aplikacji.

Folder **app** zawiera:

- folder **manifest**
- folder **java**
- folder **res** (*resources*)



Folder manifest

Zawiera plik

AndroidManifest.xml.

Każdy projekt musi posiadać ten plik. Zawiera on informacje która Aktywność (klasa) powinna zostać uruchomiona jako pierwsza, jakich zezwoleń wymaga aplikacja, jaką ma nazwę itd.



```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3         package="pl.agh.emilia.helloworld">
4
5     <application
6         android:allowBackup="true"
7         android:icon="@mipmap/ic_launcher"
8         android:label="HelloWorld"
9         android:roundIcon="@mipmap/ic_launcher_round"
10        android:supportsRtl="true"
11        android:theme="@style/AppTheme">
12        <activity android:name=".MainActivity">
13            <intent-filter>
14                <action android:name="android.intent.action.MAIN" />
15
16                <category android:name="android.intent.category.LAUNCHER" />
17            </intent-filter>
18        </activity>
19    </application>
20
21 </manifest>
```

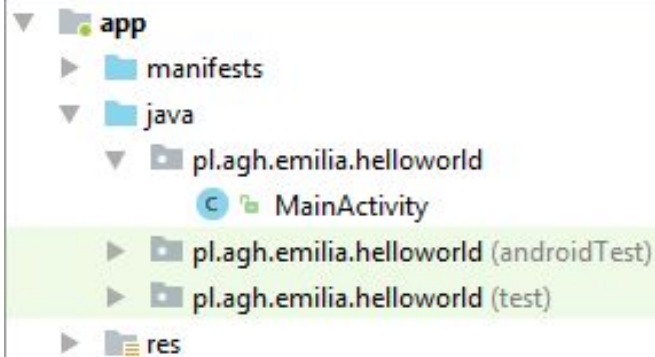


Folder java

Zawiera kod źródłowy aplikacji.

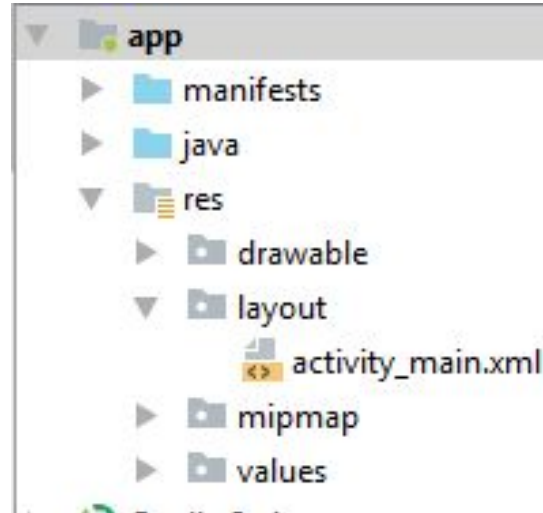
Domyślnie znajduje się tam plik **MainActivity.java**, który jest punktem wejścia aplikacji.

```
1 package pl.agh.emilia.helloworld;
2
3 import ...
4
5
6 public class MainActivity extends AppCompatActivity {
7
8     @Override
9     protected void onCreate(Bundle savedInstanceState) {
10         super.onCreate(savedInstanceState);
11         setContentView(R.layout.activity_main);
12     }
13 }
```



Folder res

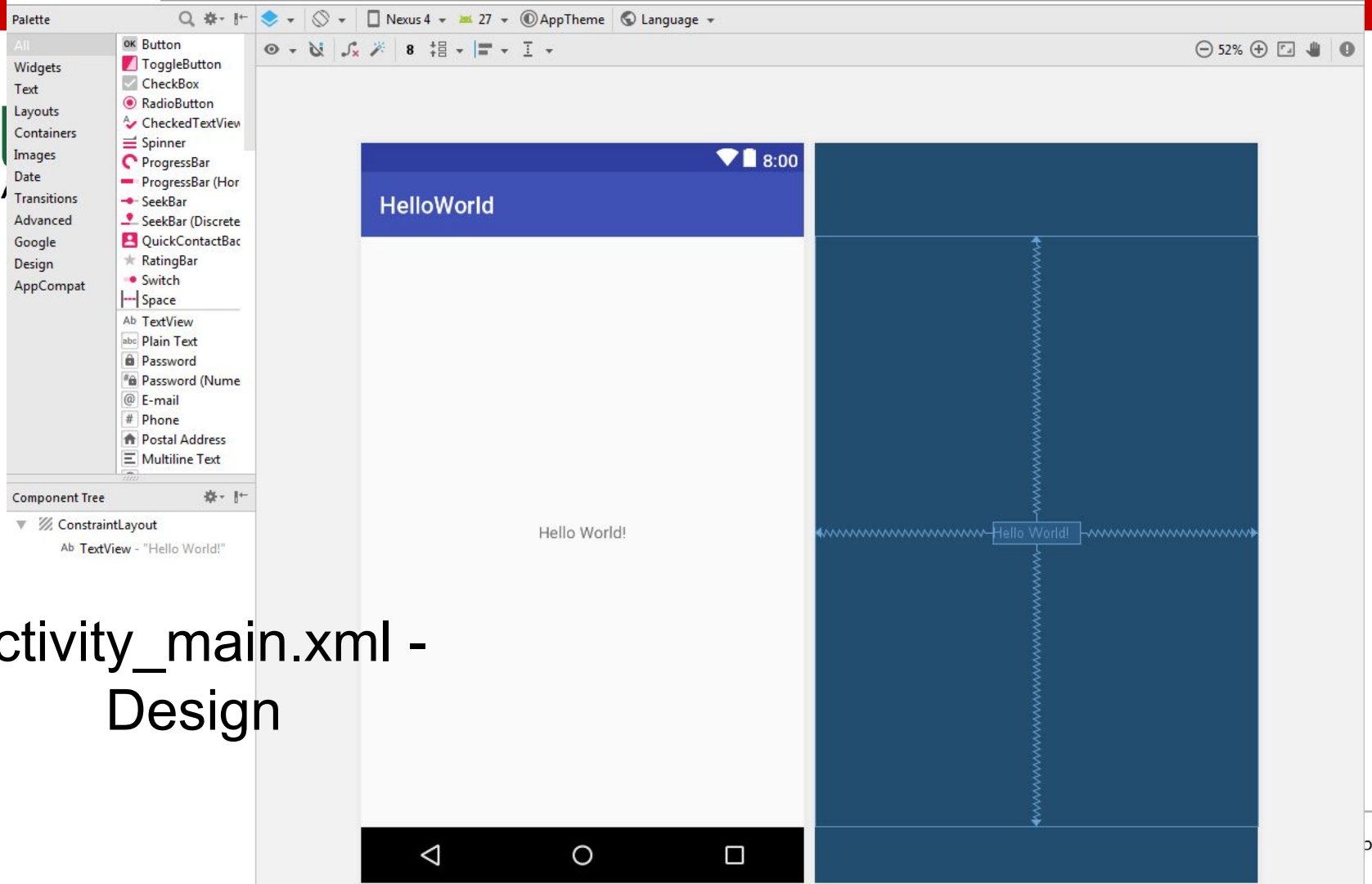
Tutaj znajduje się wszystko co można oddzielić od kodu (obrazki, animacje, layouts, dźwięki, a nawet tekst). W podfolderze **layout** znajduje się plik **activity_main.xml**, który opisuje interfejs użytkownika. Można edytować go na 2 sposoby. Bezpośrednio zmieniać kod xml lub używać wbudowanego edytora.





activity_main.xml - Text

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:layout_width="match_parent"
6     android:layout_height="match_parent"
7     tools:context="pl.agh.emilia.helloworld.MainActivity">
8
9     <TextView
10         android:layout_width="wrap_content"
11         android:layout_height="wrap_content"
12         android:text="Hello World!"
13         app:layout_constraintBottom_toBottomOf="parent"
14         app:layout_constraintLeft_toLeftOf="parent"
15         app:layout_constraintRight_toRightOf="parent"
16         app:layout_constraintTop_toTopOf="parent" />
17
18 </android.support.constraint.ConstraintLayout>
```



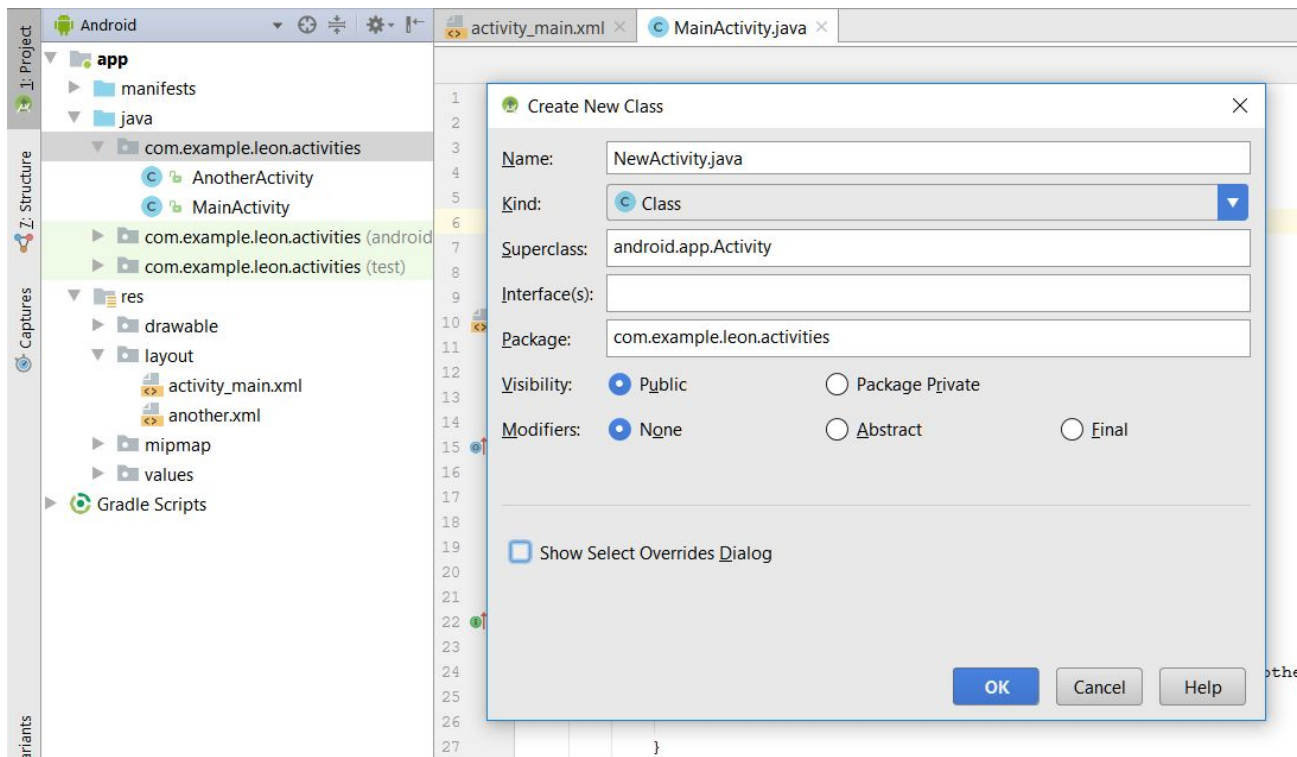
activity_main.xml -
Design

Activity (aktywność)

Klasa odpowiedzialna za interakcję z użytkownikiem, tworzenie okna naszej aplikacji i uruchamianie podstawowych komponentów systemowych.

MainActivity.java to główna aktywność.

Tworzenie aktywności

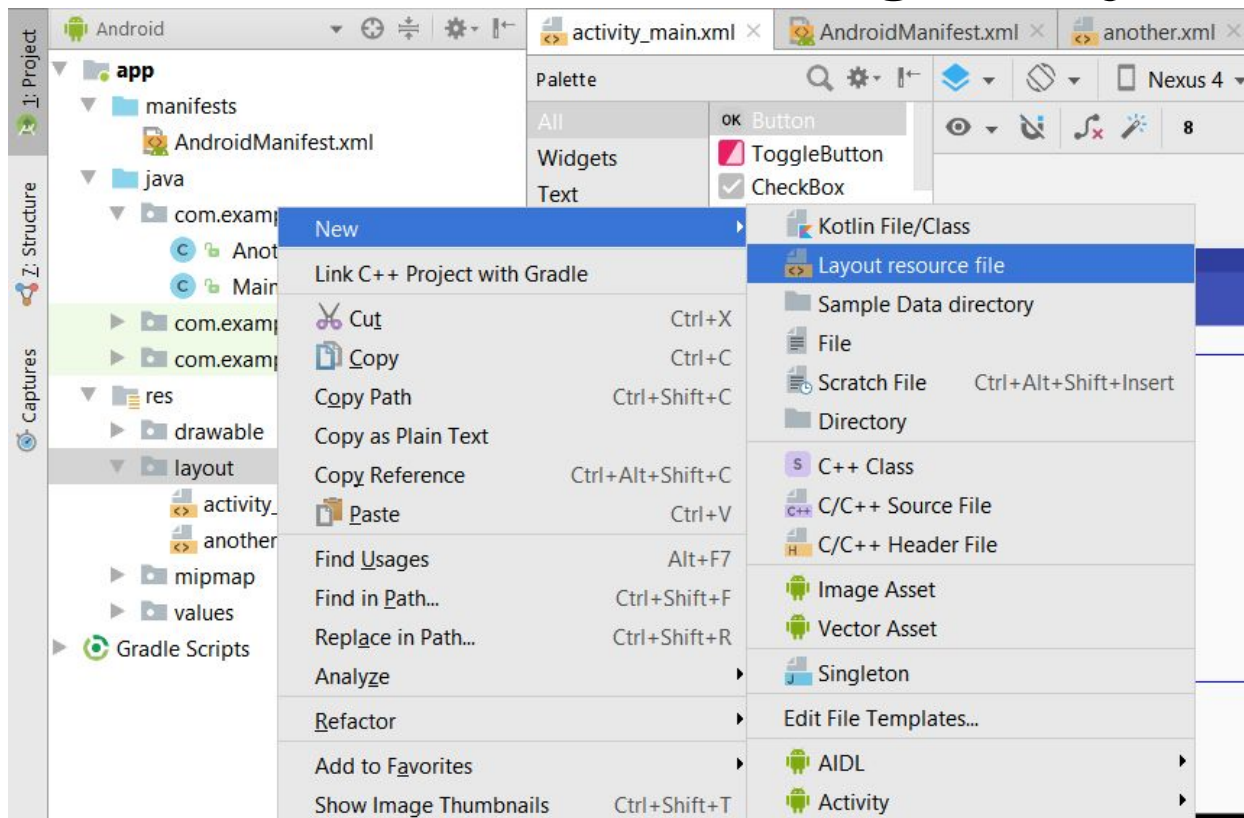


Tworzenie aktywności II

AndroidManifest.xml

```
manifest application
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3    package="com.example.leon.activities">
4
5    <application
6      android:allowBackup="true"
7      android:icon="@mipmap/ic_launcher"
8      android:label="Activities"
9      android:roundIcon="@mipmap/ic_launcher_round"
10     android:supportsRtl="true"
11     android:theme="@style/AppTheme">
12     <activity android:name=".MainActivity">
13       <intent-filter>
14         <action android:name="android.intent.action.MAIN" />
15
16         <category android:name="android.intent.category.LAUNCHER" />
17       </intent-filter>
18     </activity>
19     <activity android:name=".AnotherActivity"></activity>
20   </application>
21
22 </manifest>
```

Tworzenie nowego layout'u



Przykładowa aktywność

```
package com.example.leon.activities;

import android.app.Activity;
import android.os.Bundle;

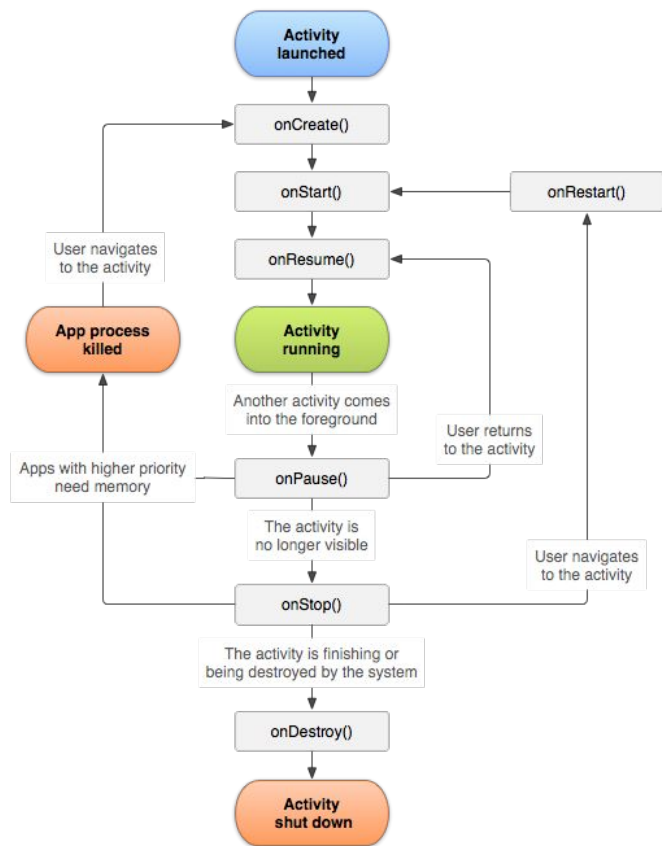
public class AnotherActivity extends Activity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.another);
    }
}
```

Stos aktywności

Wszystkie aktywności składowane są na wspólnym stosie. Im wyżej na stosie leży aktywność, tym jest nowsza i ma większy priorytet.

Kiedy zaczyna brakować zasobów, system zaczyna usuwać aktywności o najniższym priorytecie, które znajdują się na dole stosu.

Cykl życia aktywności



Cykl życia składa się z metod, które przeciążamy w naszej klasie dziedziczącej po Activity.

Cykl życia aktywności II

onCreate() - inicjalizacja aktywności, powiązanie danych z kontrolerami, tworzenie wątków itp.

onDestroy() - usunięcie wszystkich zasobów tworzonych w onCreate(), zamknięcie połączeń, np. z bazą danych

onStop() - zatrzymanie wszystkich animacji, wątków, serwisów itp, powiązanych z interfejsem aktywności

onStart()/onRestart() - przywracamy lub restartujemy wszystkie czynności/procesy zatrzymane przez funkcję onStop()

onResume(), onPause() - pomiędzy tymi metodami zawiera się **Aktywny Czas Życia**

Intents (intencje)

Służą głównie jako mechanizm do obsługi żądań użytkownika (uruchamiania aktywności). Mogą być również używane do komunikacji pomiędzy aplikacjami lub mniejszymi komponentami.

Typy uruchomień

- jawne (explicit) - jawnie deklarujemy obiekt, który chcemy stworzyć

```
Intent intent = new Intent (getApplicationContext(), AnotherActivity.class);
```

- niejawne (implicit) - zawieramy informacje, co chcemy zrobić i na jakich danych, ale nie mówimy wprost, kto ma to zrobić. Można nimi sterować za pomocą filtrów intencji

```
Uri url = Uri.parse("http://www.google.com");  
Intent intent = new Intent(Intent.ACTION_VIEW, url);  
startActivity(intent);
```

Uruchamianie aktywności

Głównym celem intencji jest zwykle tworzenie aktywności. Przełączamy na nią:

```
Intent intent= new Intent(getApplicationContext(), AnotherActivity.class);  
startActivity(intent);
```

Przesyłanie informacji

Aby dołączyć do wywołania dane należy użyć funkcji `putExtra`:

```
intent.putExtra("UserID", 123);
```

Odbieramy dane w wywołanej aktywności:

```
Intent intent = getIntent();  
int userId = intent.getIntExtra("UserID", 0);  
//Drugi argument funkcji getIntExtra to wartość  
domyślna, na wypadek gdyby nie została nam wcześniej  
przekazana.
```

Zadanie

Aplikacja zmieniająca kolor tła po naciśnięciu przycisku.



activity_main.xml

```
1 <?xml version="1.0" encoding="utf-8"?>
2 <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
3     xmlns:app="http://schemas.android.com/apk/res-auto"
4     xmlns:tools="http://schemas.android.com/tools"
5     android:id="@+id/layout"
6     android:layout_width="match_parent"
7     android:layout_height="match_parent"
8     tools:context="pl.agh.emilia.backgroundchange.MainActivity">
9
10     <Button
11         android:id="@+id/button"
12         android:layout_width="wrap_content"
13         android:layout_height="wrap_content"
14         android:layout_centerInParent="true"
15         android:text="Change" />
16
17
18 </RelativeLayout>
```



AGU

MainActivity.java

```
public class MainActivity extends AppCompatActivity {  
    Button button;  
    RelativeLayout layout;  
  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
  
        button = (Button) findViewById(R.id.button);  
  
        layout = (RelativeLayout) findViewById(R.id.layout);  
  
        button.setOnClickListener(new View.OnClickListener() {  
            @Override  
            public void onClick(View v) {  
                Random rnd = new Random();  
                int color = Color.argb( alpha: 255, rnd.nextInt( bound: 256), rnd.nextInt( bound: 256), rnd.nextInt( bound: 256));  
                layout.setBackgroundColor(color);  
            }  
        });  
    }  
}
```