

Übung 1 – Activities und Intents

In unserer allerersten Übung werden wir natürlich das Android SDK, die IDE Android Studio sowie den Emulator und – falls vorhanden – euer Android-Gerät installieren. In einem zweiten Schritt geht es dann darum, den Activity-Lebenszyklus genauer anzuschauen und Activities mit Intents zu starten.

Die Ziele für die Übung sind:

- ★ Sie können sich die nötige Umgebung für die Android-Entwicklung einrichten.
- ★ Sie haben Ihre erste Android App erstellt!
- ★ Sie verstehen den Activity-Lebenszyklus und wissen, welche Aktionen im jeweiligen Schritt gemacht werden sollten.

Das Online-Übungsblatt mit aktualisierten Hinweisen finden Sie unter <https://goo.gl/GXbNoq>.

Aufgabe 1 – Setup

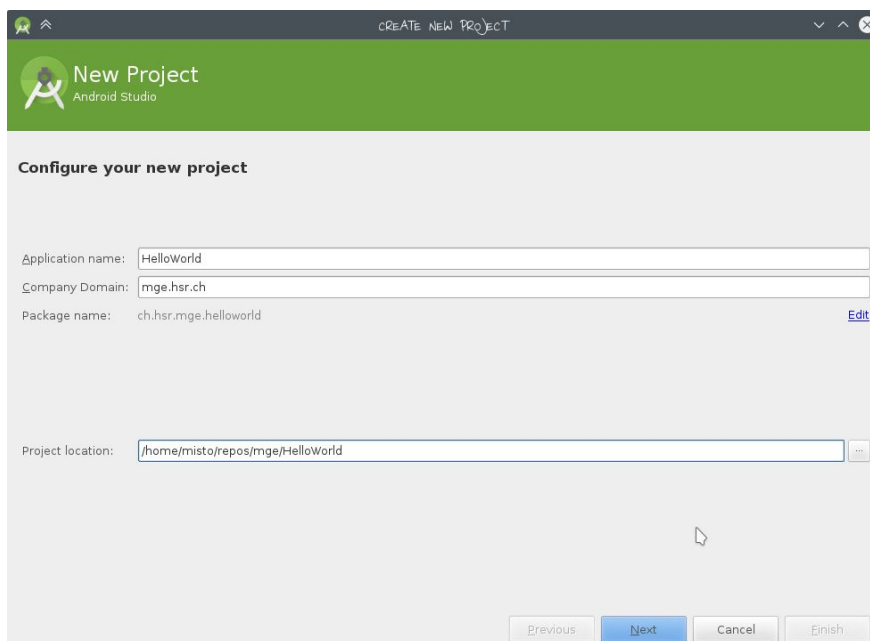
Die Installation ist je nach bereits installierten Komponenten und verwendetem Betriebssystem unterschiedlich und enthält verschiedene Stolpersteine. Folgen Sie für die Installation dieser sehr detaillierten und umfassenden Anleitung:

<http://www.programmierenlernenhq.de/tutorial-android-studio-2-installieren/>


Folgen Sie der Anleitung bis ans Ende, bis Sie das Android-Studio und die empfohlenen Pakete des SDKs (API Level 23 oder 24, je nach Gerät) installiert haben. Falls Sie OSX einsetzen ist der letzte Schritt *Installation von Intel HAXM* bei [Intel](#) gut erklärt. Unter Linux brauchen Sie bloss [KVM installiert](#) zu haben. Falls Sie nicht vorhaben, den Emulator zu benutzen, können Sie diesen Schritt natürlich überspringen.

Aufgabe 2 – Hello World

Starten Sie Android Studio und erstellen Sie mit dem Wizard ein neues Projekt. Starten Sie dabei mit einer leeren Activity, und wählen Sie API-Level 14.



Hinweis: Wählen Sie keinen zu hohen API-Level, auch wenn Sie die neuste Android-Version auf Ihrem Gerät/Emulator haben, da wir bewusst die Kompatibilitätslibraries einsetzen werden. Level 19 ist ein guter Kompromiss zwischen Features und Verbreitung.

Nach einigen Sekunden sollte die App kompilieren und ausführbar sein. Mit  erscheint ein Auswahldialog für die erkannten Smartphones und Emulatoren:

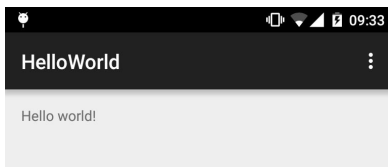
App auf dem Smartphone starten

Um die App auf dem Smartphone zu starten, müssen Sie das Gerät zuerst in den Entwicklermodus versetzen und allenfalls noch einen Treiber installieren. Eine detaillierte Anleitung ist wieder im [Programmieren Lernen HQ](#) zu finden.

App im Emulator starten

Beim ersten Starten ist noch kein emuliertes Gerät vorhanden, Sie müssen dieses also erst noch erstellen. Folgen Sie den Voreinstellungen oder stellen Sie sich Ihr Wunschgerät zusammen und starten Sie die App danach.

Ob Sie die App auf dem Smartphone oder im Emulator starten, folgendes Resultat sollten Sie am Ende erhalten:



Sie sind nun bereit, im nächsten Schritt eine etwas interessantere App zu schreiben.

Aufgabe 3 – Activity-Lifecycle

Um ein besseres Verständnis für den Activity-Lifecycle zu bekommen erstellen wir nun eine App, die uns die jeweiligen Zustände mitteilt. Erstellen Sie dazu eine neue App, und überschreiben Sie in der Activity-Klasse die Lifecycle-Methoden die Sie in der Vorlesung kennengelernt haben.

Zudem brauchen wir eine Möglichkeit die Zustandsänderungen zu kommunizieren. Wir können dies mit der Log-Konsole tun (System.out funktioniert unter Android nicht) oder über Notifications. Die folgende Methode implementiert beide Varianten – Achtung, beim Kopieren aus dem PDF kann es passieren, dass unerwartete Zeichen (Spaces, Bindestriche) im Sourcecode landen, am sichersten ist es, wenn der Code aus dem Google Doc kopiert wird:

```
private void stateChangedTo(String state) {
    String currentTimeString = getCurrentTimeString();

    Notification notification =
        new Notification.Builder(this)
            .setContentTitle(state)
            .setSmallIcon(android.R.drawable.presence_busy)
            .setContentText(currentTimeString)
            .getNotification();
    NotificationManager notificationManager =
```

```
(NotificationManager) getSystemService(NOTIFICATION_SERVICE);  
notificationManager.notify((int) System.currentTimeMillis(), notification);  
  
Log.d(getString(R.string.app_name), state + " at " + currentTimeString);  
}  
  
private String getCurrentTimeString() {  
    return new SimpleDateFormat("HH:mm:ss.SSS").format(new Date());  
}
```

Rufen Sie also die `stateChangedTo` Methode in den überschriebenen Activity-Methoden auf und starten Sie die App.

Welche Methoden werden beim Starten der App aufgerufen?

`onCreate`, `onStart`, `onResume`

Was geschieht, wenn der Screen gesperrt wird?

`onPause`, `onStop`

Und beim Entsperren?

`onStart`, `onResume`

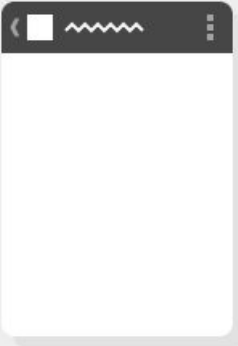
Wann wird `onDestroy` ausgeführt?

Falls "finish" auf der Activity aufgerufen wird - oder wei sie zum Speicher Spaaren vom System gekillt wird.

Aufgabe 4 – Activity-Lifecycle Erweitert mit Intents

Erweitern Sie die App nun um eine zweite Activity:

Creates a new blank activity with an action bar.



Blank Activity

Activity Name:	<input type="text" value="SecondActivity"/>
Layout Name:	<input type="text" value="activity_second"/>
Title:	<input type="text" value="SecondActivity"/>
Menu Resource Name:	<input type="text" value="menu_second"/>
	<input type="checkbox"/> Launcher Activity
Hierarchical Parent:	<input type="text" value="ch.hsr.mge.activitylifecycleintents.MainActivity"/> ...
Package name:	<input type="text" value="ch.hsr.mge.activitylifecycleintents"/>

Da unsere Android-Programmierkünste leider noch nicht so ausgereift sind, begnügen wir uns damit, in der MainActivity den Screen mit einem Button zu füllen, und beim Drücken die zweite Activity zu starten. Ergänzen Sie `onCreate` mit:

```
Button button = new Button(this);
button.setText("Launch!");
setContentView(button);

button.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        // TODO
    }
});
```

Welche Zustände werden beim Aufrufen und Verlassen der SecondActivity durchlaufen?

Aufgabe 5 – Launch Modes (optional)

Falls Sie das Thema der Launch Modes noch ein wenig vertiefen und besser verstehen wollen laden Sie die [Activities LaunchMode Demo](#) App herunter ([Code ist auf GitHub](#)).