JKU Young Computer Scientists

Mobile Computing

JOHANNES KEPLER
UNIVERSITÄT LINZ

Matthias Steinbauer
matthias.steinbauer@jku.at
@climbinggeek

Matthias Steinbauer

Studium Informatik JKU, derzeit PhD am
Institut für Telekooperation
(Big Data, Temporal Graphs)

Seit 2012 Universitätsassistent @JKU
- 8 angeleitete Bachelor- und Masterarbeiten
- 15 wissenschaftliche Publikationen

Seit 2004 Selbstständig
- Softwareentwicklung Java EE Umfeld
- Trainings Prozessmodellierung
- Beratung Big Data, Prozesse, Java EE

JOHANNES KEPLER UNIVERSITÄT LINZ

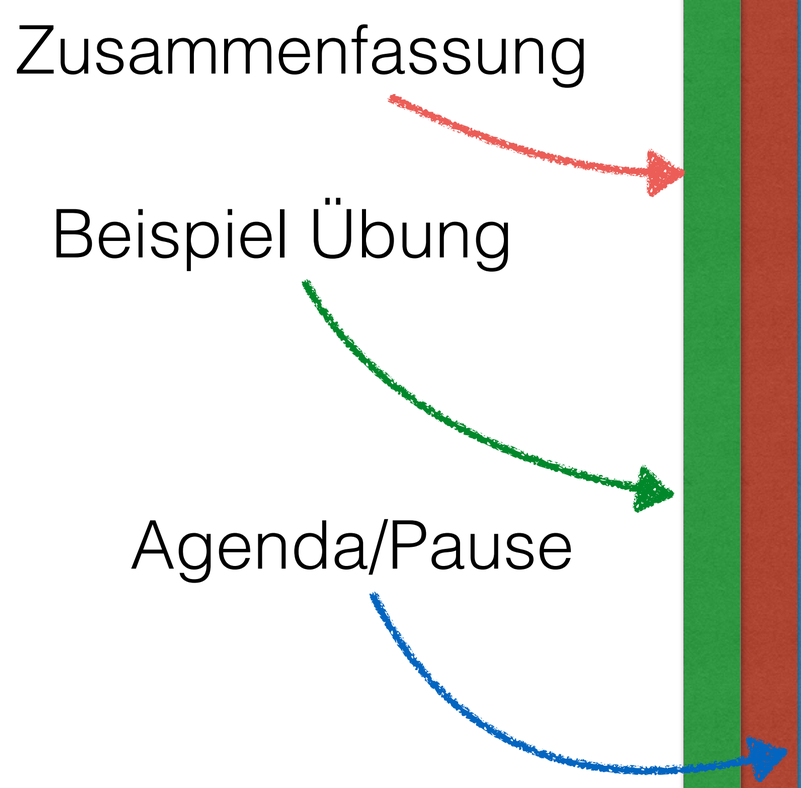
Raiffeisen

Immformer
Zeit fürs Wesentliche

IDS SCHEER CONSULTING

matthias@steinbauer.org
http://steinbauer.org/

Hinweise zum Slide-Deck

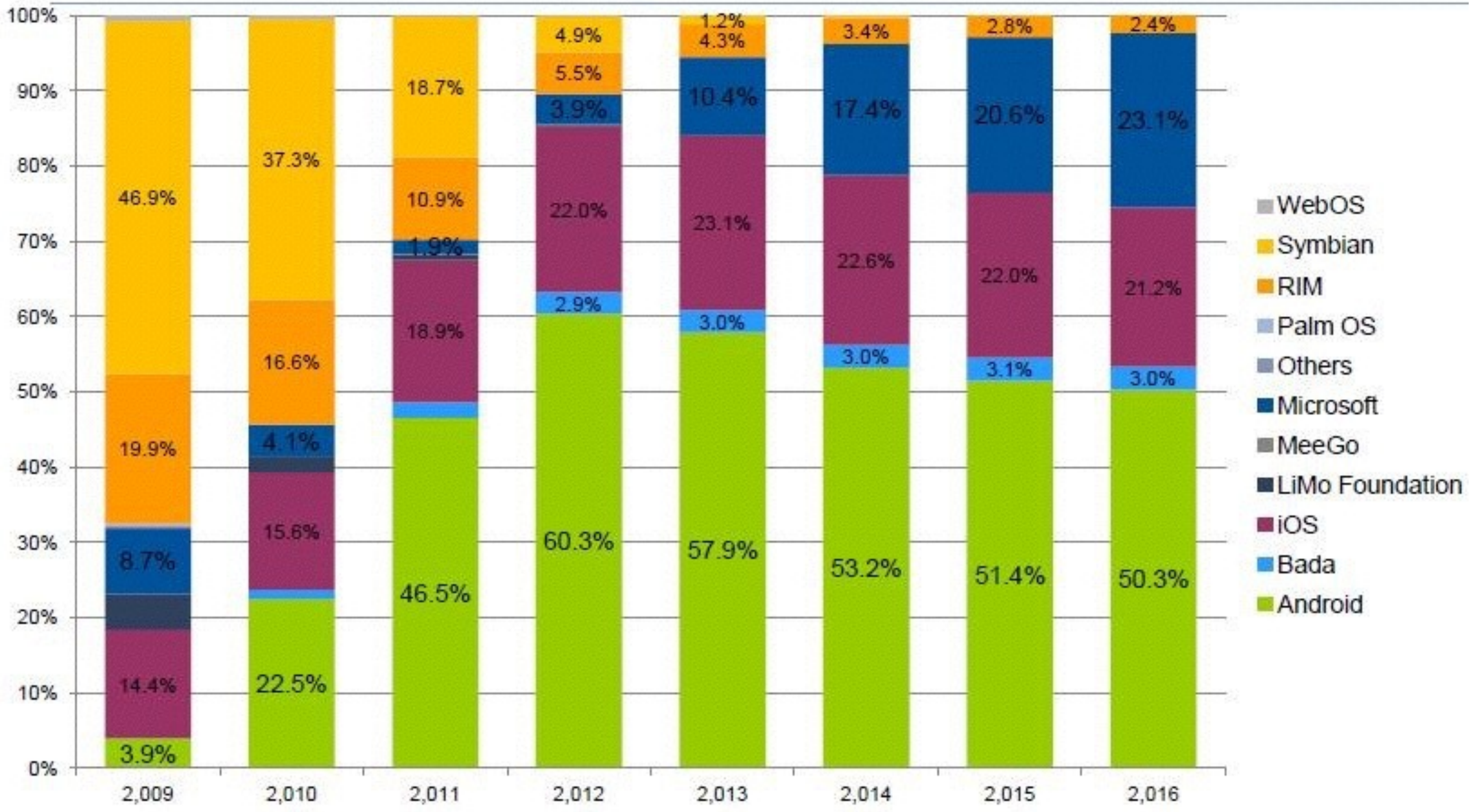


Heutige Aufgaben

- Kennenlernen von Android
- Erstellen eines Projektes in Android Studio
- Zusammensetzen eines UI
- Anbinden von Handler Funktionen
- Erstellen eines TicTacToe Games
- Internet Multiplayer Fähigkeit

Was ist Android?

- **Betriebssystem** für Smartphones von **Google** entwickelt
- Enthält Software Development Kit (**SDK**) welches das Erstellen von **Apps** in der Programmiersprache **Java** erlaubt
- **Entwicklungsumgebung** ist **gratis** verfügbar
- Ist auf ca. **50%** aller Geräte die im Umlauf sind installiert



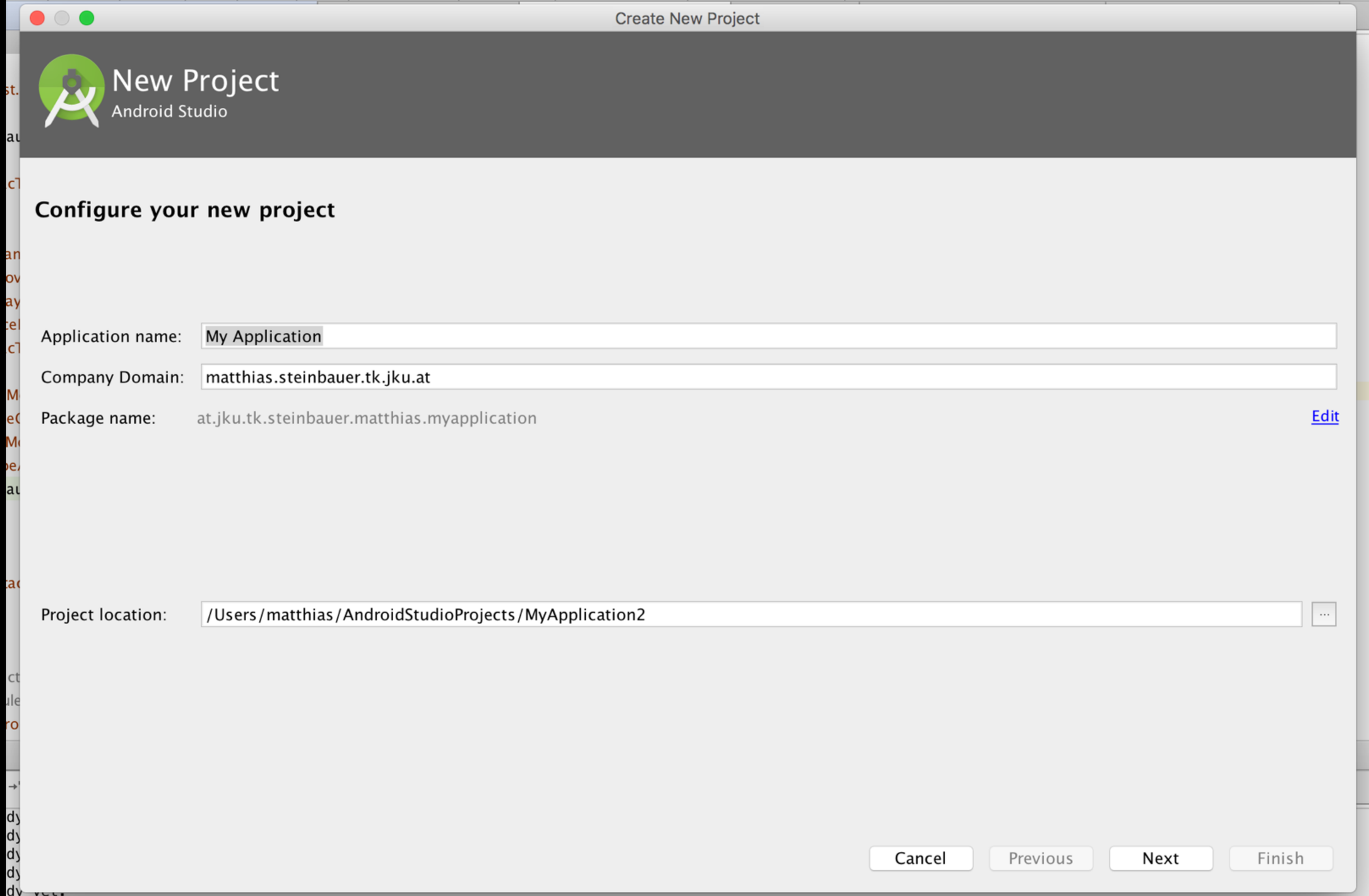
Source: Gartner

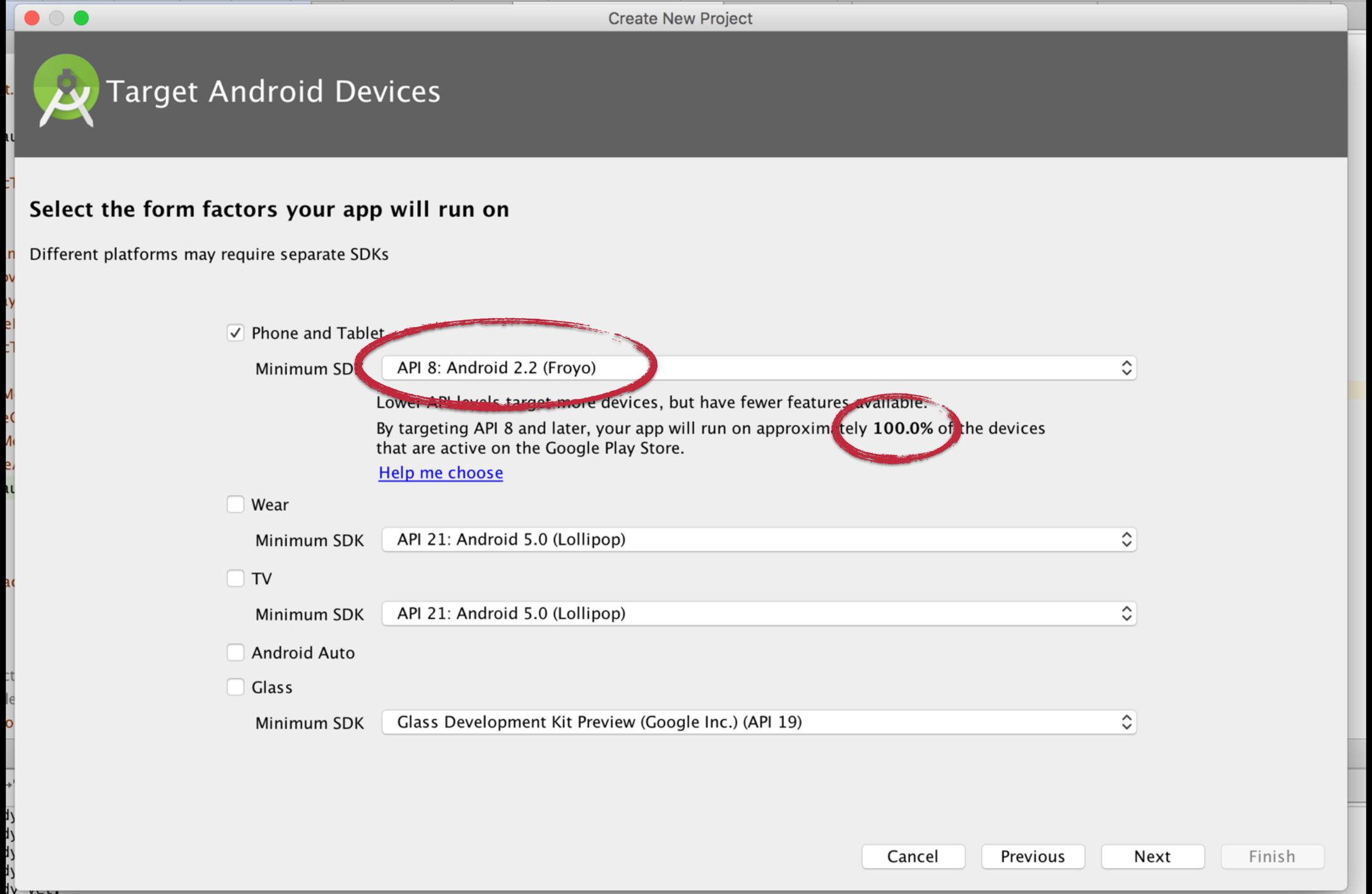
Forecast: Mobile Devices by Open Operating System, Worldwide, 2009-2016, 2Q12 Update

Gartner

Android Studio Starten

- **Android Studio** starten und ein **Projekt einrichten**
- **Android Emulator (AVD)** starten und virtuelles Gerät einrichten
- Das leere Projekt starten und schauen was in der Entwicklungsumgebung passiert

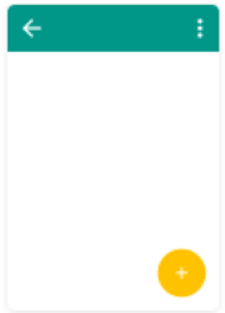

Create New Project
New Project
Android Studio
Configure your new project
Application name:
My Application
Company Domain:
matthias.steinbauer.tk.jku.at
Package name:
at.jku.tk.steinbauer.matthias.myapplication
Edit
Project location:
/Users/matthias/AndroidStudioProjects/MyApplication2
...
Cancel
Previous
Next
Finish


Create New Project
Target Android Devices
Select the form factors your app will run on
Different platforms may require separate SDKs
Phone and Tablet
Minimum SDK
API 8: Android 2.2 (Froyo)
devices, but have fewer features available.
By targeting API 8 and later, your app will run on approximately 100.0% of the devices that are active on the Google Play Store.
Help me choose
Wear
Minimum SDK
API 21: Android 5.0 (Lollipop)
TV
Minimum SDK
API 21: Android 5.0 (Lollipop)
Android Auto
Glass
Minimum SDK
Glass Development Kit Preview (Google Inc.) (API 19)
Cancel
Previous
Next
Finish

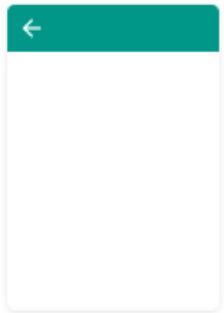
Create New Project

Add an activity to Mobile

Add No Activity

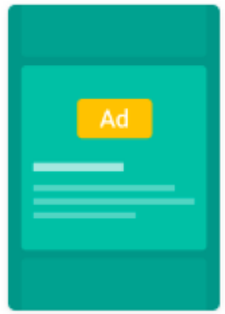


Blank Activity

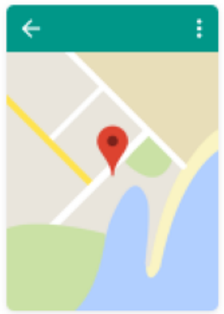


Empty Activity

Fullscreen Activity

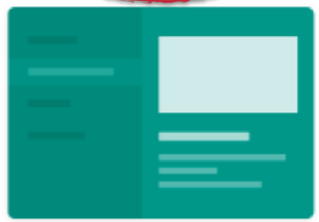


Google AdMob Ads Activity

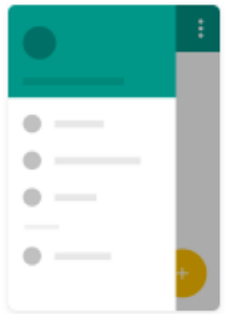


Google Maps Activity

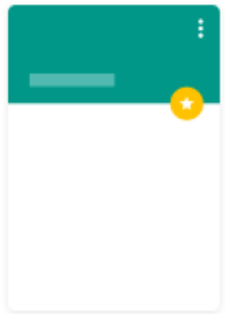
Login Activity



Master/Detail Flow



Navigation Drawer Activity



Scrolling Activity

Cancel | Previous | Next | Finish

Kompatiblität zu aktuellen Android Versionen herstellen

- In der Datei
  `build.gradle (Module: app)`
  kontrollieren ob folgende Zeile eingetragen ist
  (Block **`dependencies`**)

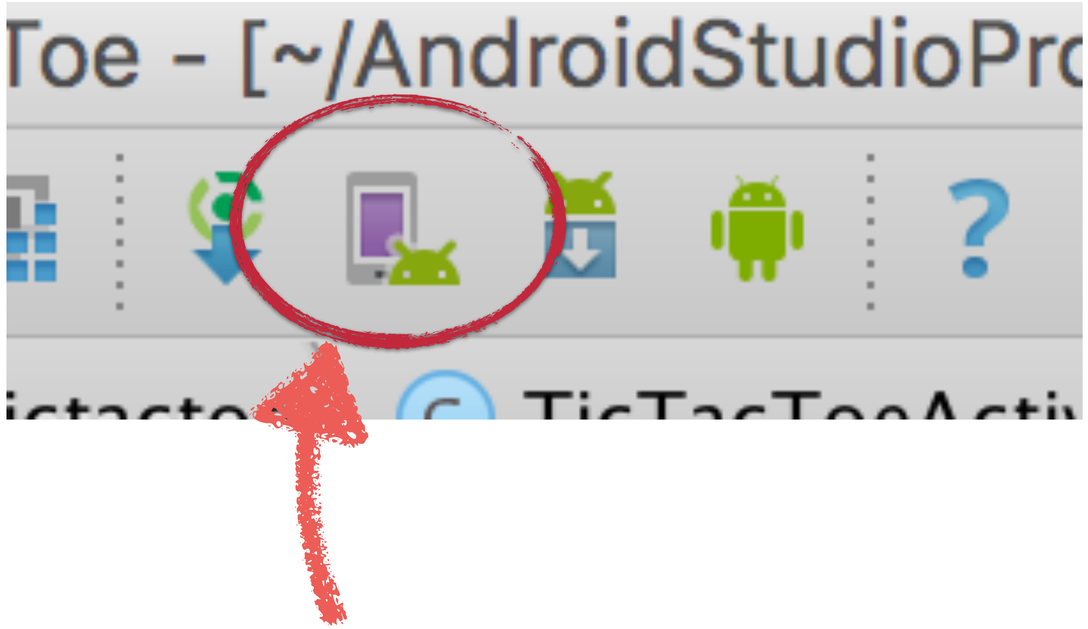
```
compile 'com.android.support:appcompat-v7:23.2.0'
```

Android Studio Starten

- **Android Studio** starten und ein **Projekt einrichten**
- **Android Emulator (AVD)** starten und virtuelles Gerät einrichten
- Das leere Projekt starten und schauen was in der Entwicklungsumgebung passiert

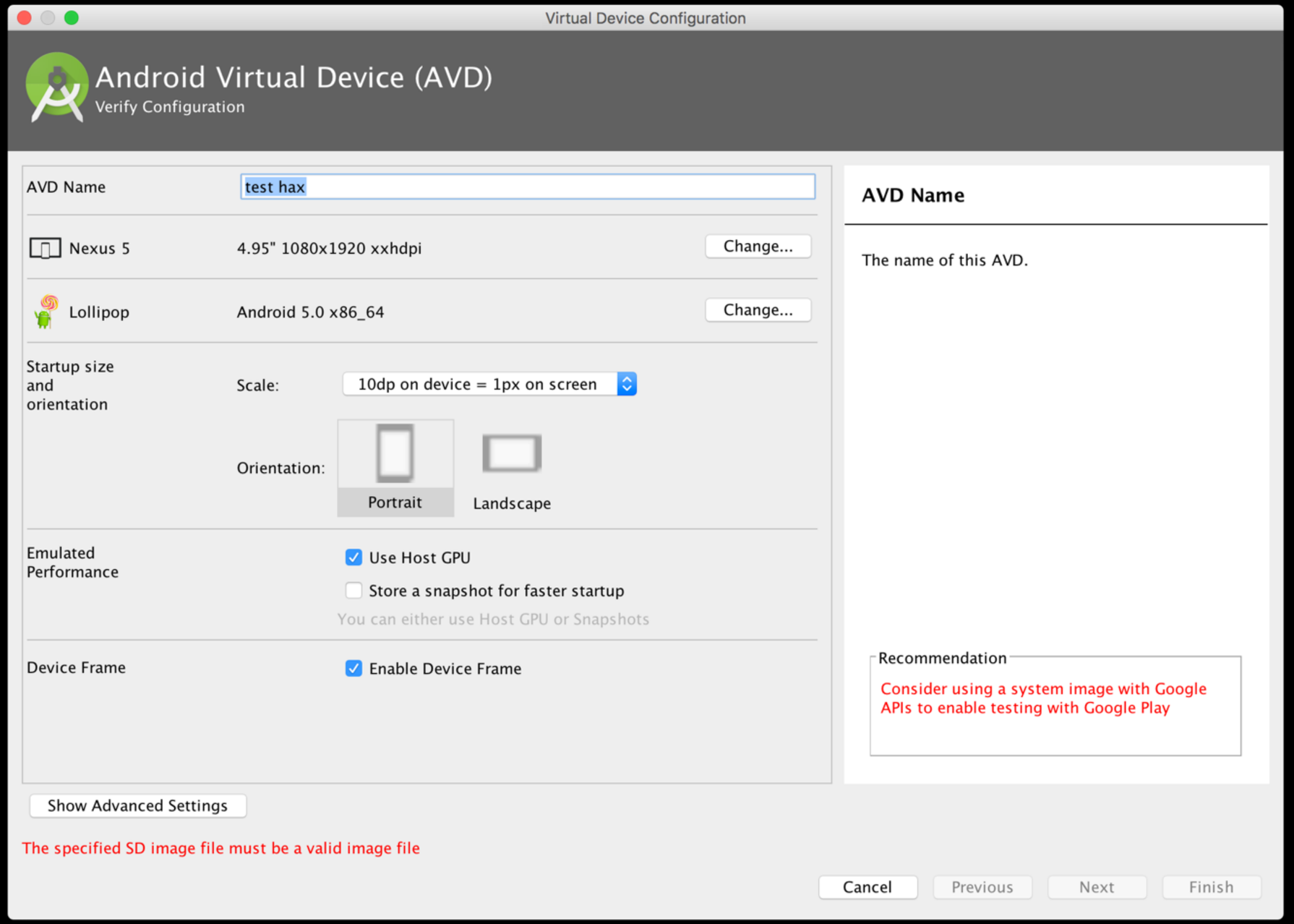
Virtuelles Device anlegen

- Emulator Gerät zum Entwickeln einrichten
- Intel x86 und ARM basierte Emulationen



AVD Manager


Android Virtual Device Manager
Your Virtual Devices
Android Studio
Type
Name
Resolution
API
Target
CPU/ABI
Size on Disk
Actions
Nexus...
768 ×...
22
Androi...
x86...
1 GB
test hax
1080 ...
21
Androi...
x86...
1 GB
Create Virtual Device...

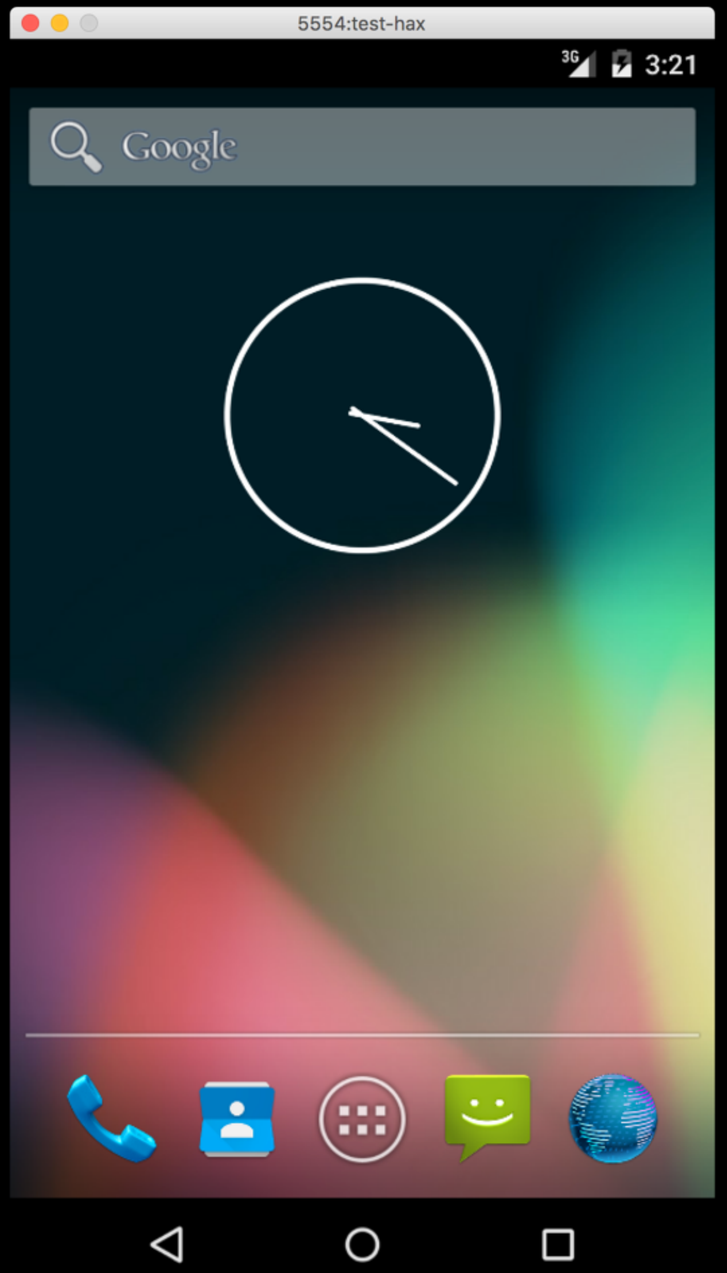


Nexus 5 oder 4

x86 bzw. x86_64
auf neuen Rechnern
mit Intel CPU

ARM um auf
der sicheren Seite
zu sein

Use Host GPU
für schnelle Grafik



Emulator fährt ein Android Betriebssystem mit allem was dazugehört

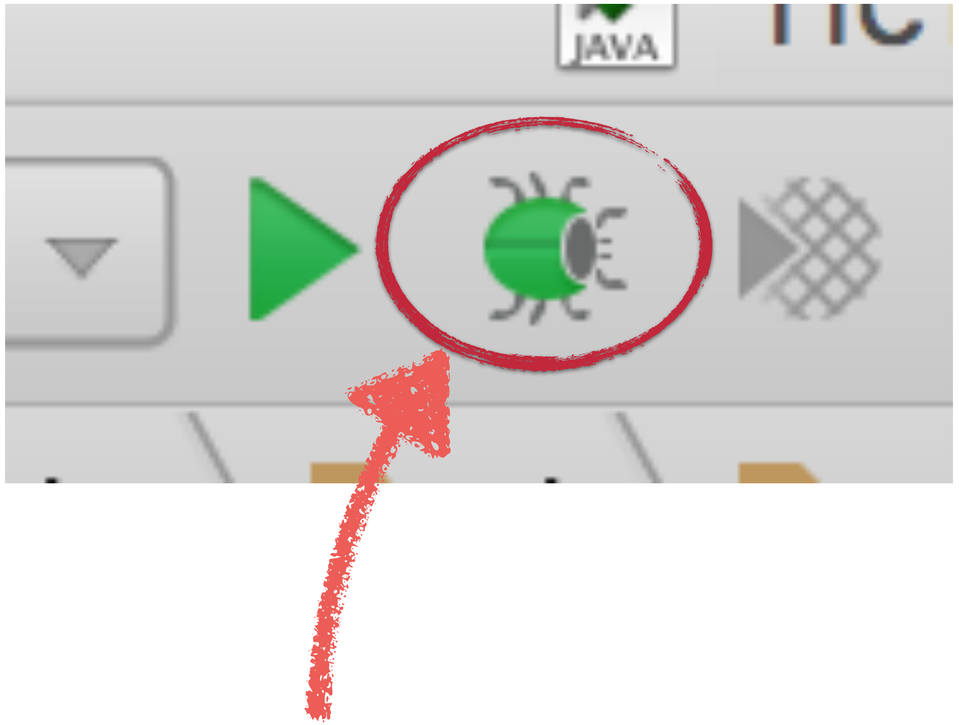
Internet Browser, etc. sollte funktionieren > Testen!

Android Studio Starten

- **Android Studio** starten und ein **Projekt einrichten**
- **Android Emulator (AVD)** starten und virtuelles Gerät einrichten
- Das leere Projekt starten und schauen was in der Entwicklungsumgebung passiert

Projekt Starten

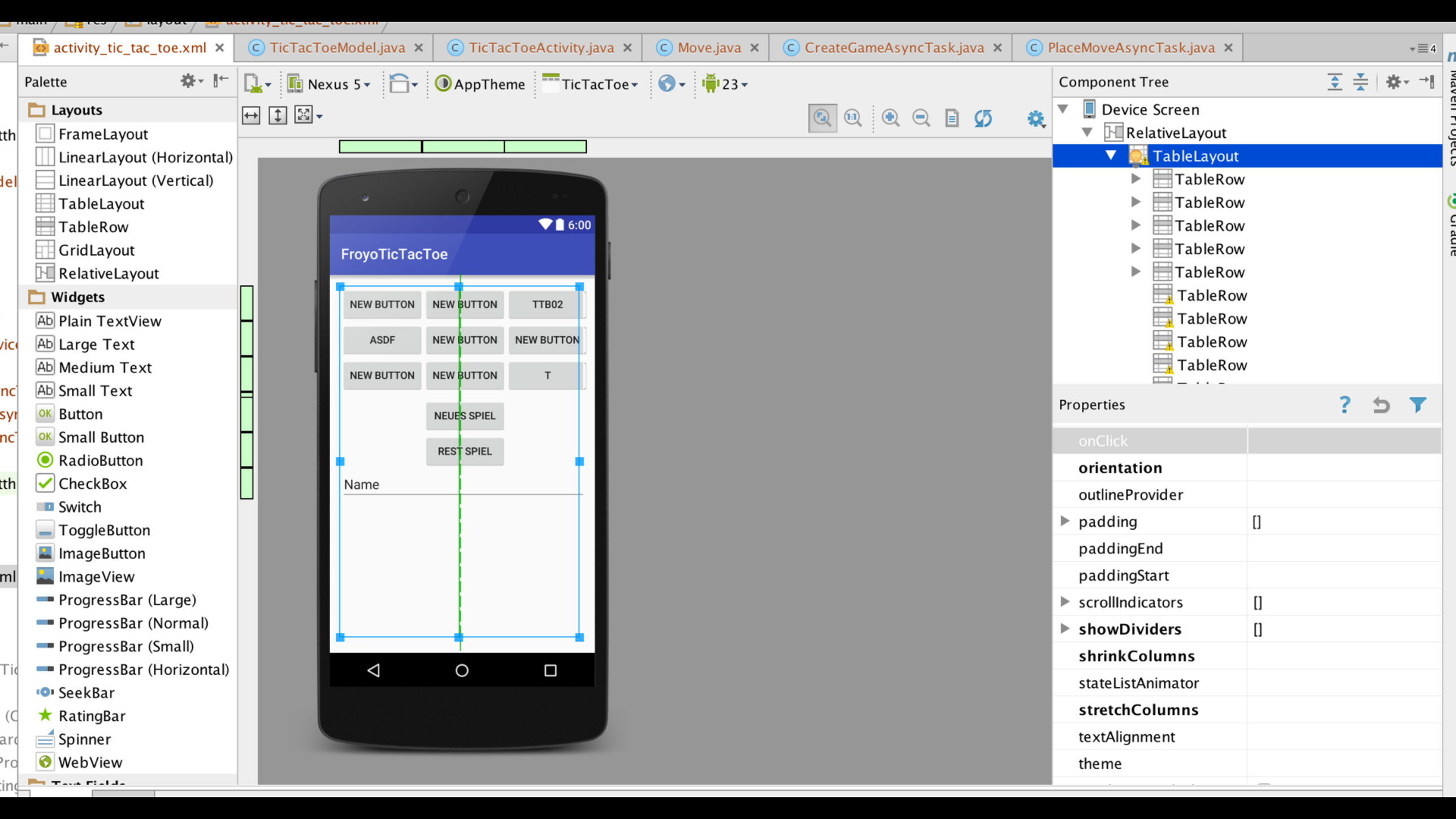
- Runtime vs. Debug Modus
- Das Projekt sollte sich ohne Problem starten lassen und eine leere Android UI anzeigen



Debug Modus

UI Zusammensetzen

- In Android Studio ist ein **UI Framework** integriert
- Öffnen von `activity_tic_tac_toe.xml`
- `TableLayout` in die View ziehen
- 9 `Button` Widgets für ein Tic Tac Toe Game in die View ziehen
- Ein weiterer `Button` für “Neues Spiel”

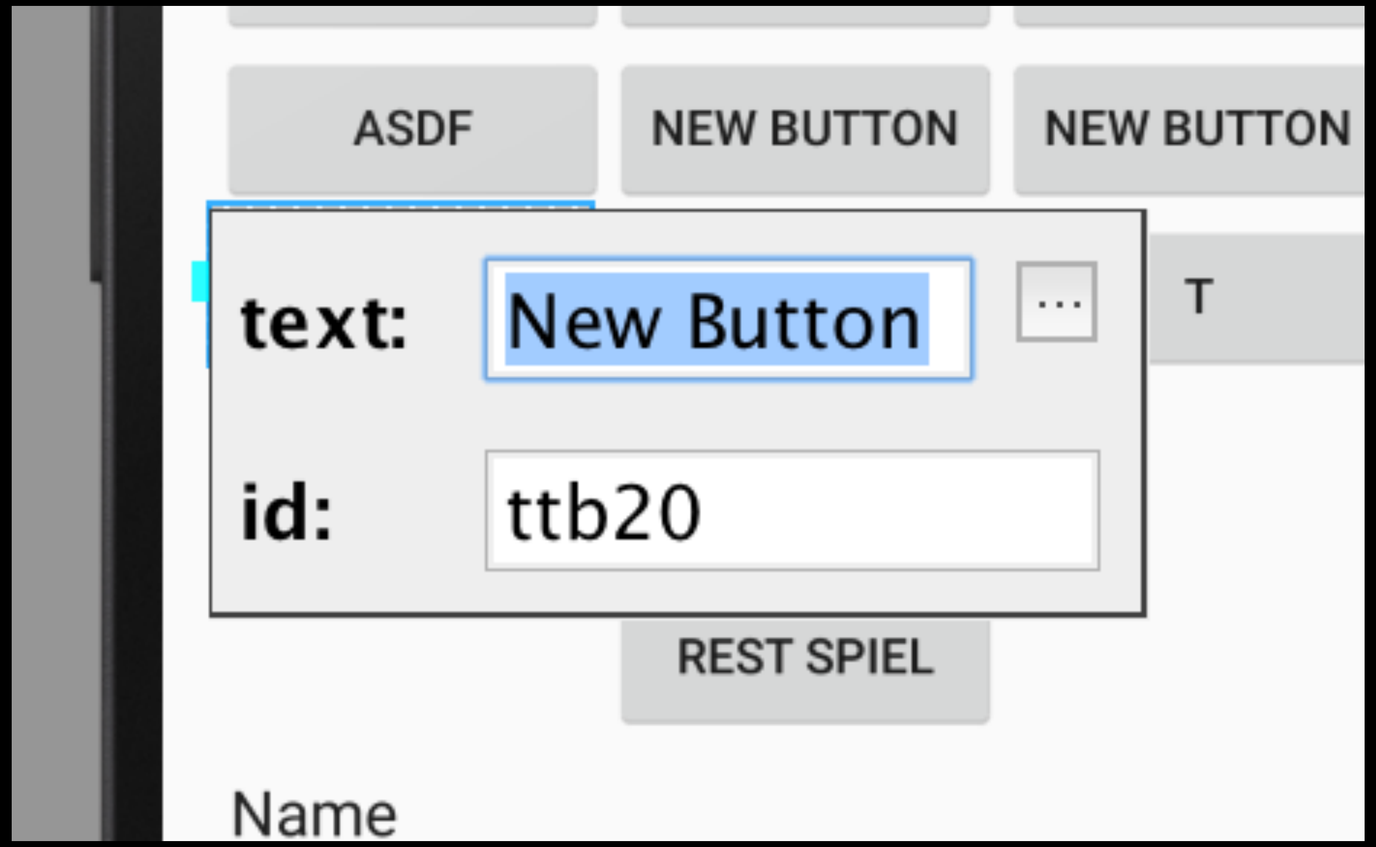

activity_tic_tac_toe.xml
TicTacToeModel.java
TicTacToeActivity.java
Move.java
CreateGameAsyncTask.java
PlaceMoveAsyncTask.java
Palette
Layouts
FrameLayout
LinearLayout (Horizontal)
LinearLayout (Vertical)
TableLayout
TableRow
GridLayout
RelativeLayout
Widgets
Plain TextView
Large Text
Medium Text
Small Text
Button
Small Button
RadioButton
CheckBox
Switch
ToggleButton
ImageButton
ImageView
ProgressBar (Large)
ProgressBar (Normal)
ProgressBar (Small)
ProgressBar (Horizontal)
SeekBar
RatingBar
Spinner
WebView
Nexus 5
AppTheme
TicTacToe
23
6:00
FroyoTicTacToe
NEW BUTTON
NEW BUTTON
TTB02
ASDF
NEW BUTTON
NEW BUTTON
NEW BUTTON
NEW BUTTON
T
NEUES SPIEL
REST SPIEL
Name
Component Tree
Device Screen
RelativeLayout
TableLayout
TableRow
TableRow
TableRow
TableRow
TableRow
TableRow
TableRow
TableRow
TableRow
Properties
onClick
orientation
outlineProvider
padding []
paddingEnd
paddingStart
scrollIndicators []
showDividers []
shrinkColumns
stateListAnimator
stretchColumns
textAlignment
theme

Doppelklick auf Element
öffnet die wichtigsten
Eigenschaften

Text beliebig bzw.
“Neues Spiel”

ids für das Spielfeld
ttb<zeile><spalte>

ttb00 (links oben)
ttb22 (rechts unten)
ttb20 (letze Zeile, erste Spalte)

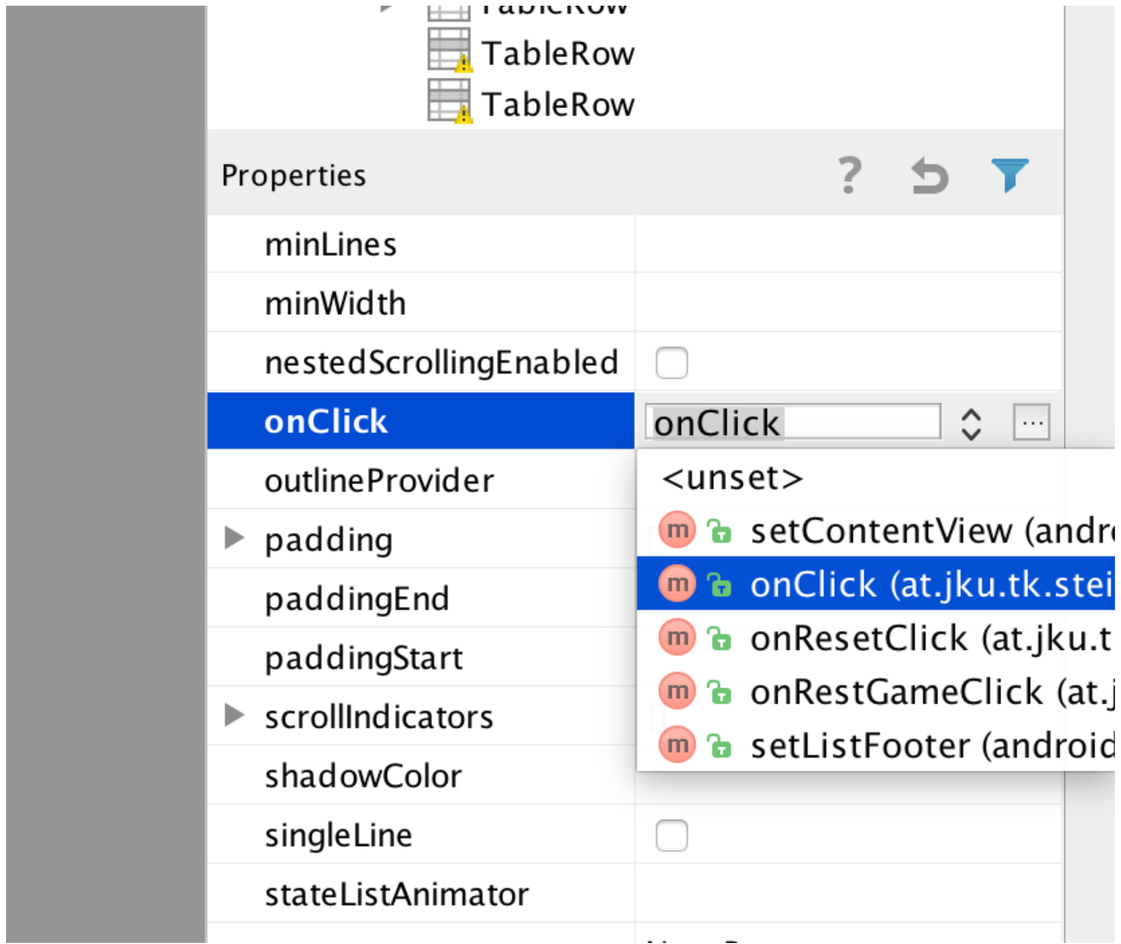


Im File **`TicTacToeActivity.java`** eine **`onClick`** Handler Methode anlegen

`TicTacToeActivity.java`

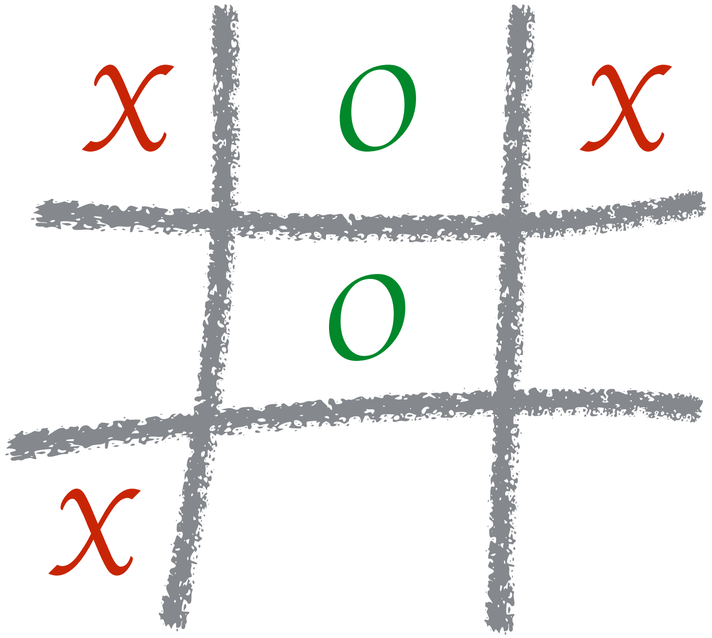
```
public void onClick(View view) {
  Log.d("onClick",
    "Ein Button wurde geklickt");
}
```

Methode als onClick Handler für alle Spielfeld Schaltflächen setzen



Speichermodell für Spiel anlegen

- Wie schaut das Spielfeld aus?
- Welche Elemente hat es?
- Wie können diese in Software dargestellt werden?



TicTacToeModel.java

```
public class TicTacToeModel {
  public static final int GAME_SIZE = 3;
  public enum TTTMark {
    Tic, Tac, None
  }
  private TTTMark currentPlayer;
  private TTTMark[][] field = new TTTMark[GAME_SIZE][GAME_SIZE];
}
```

Mögliche Spielfeld-Belegungen

Wer ist dran?

Spielfeld

Konstante für Spielfeld-Größe

https://github.com/steima/FroyoTicTacToe

TicTacToeModel.java

Constructor
wird beim anlegen
des Objekts
ausgeführt

Können wir aufrufen
um ein neues Spiel
zu beginnen

```java
public TicTacToeModel() {
  this.resetField();
  this.currentPlayer = TTTMark.Tic;
}

public void resetField() {
  for(int row=0;row<GAME_SIZE;row++) {
    for(int col=0;col<GAME_SIZE;col++) {
      this.field[row][col] = TTTMark.None;
    }
  }
}
```

TicTacToeActivity.java

```
private TicTacToeModel gameModel;

private Button[][] gameButtons = new
Button[TicTacToeModel.GAME_SIZE]
[TicTacToeModel.GAME_SIZE];

private void updateGameButtonReferences() {
  this.gameButtons[0][0] = (Button)
findViewById(R.id.ttb00);
  this.gameButtons[0][1] = (Button)
findViewById(R.id.ttb01);
  this.gameButtons[0][2] = (Button)
findViewById(R.id.ttb02);
  this.gameButtons[1][0] = (Button)
findViewById(R.id.ttb10);
  this.gameButtons[1][1] = (Button)
findViewById(R.id.ttb11);
  this.gameButtons[1][2] = (Button)
findViewById(R.id.ttb12);
  this.gameButtons[2][0] = (Button)
findViewById(R.id.ttb20);
  this.gameButtons[2][1] = (Button)
findViewById(R.id.ttb21);
  this.gameButtons[2][2] = (Button)
findViewById(R.id.ttb22);
}
```

```
public void updateUiWithModel() {
  this.updateGameButtonReferences();
  for(int row=0;
      row<TicTacToeModel.GAME_SIZE;row++) {
    for(int col=0;
        col<TicTacToeModel.GAME_SIZE;col++) {
      TicTacToeModel.TTTMark tmp =
         gameModel.getMark(row, col);
      switch (gameModel.getMark(row, col)) {
        case None:
          gameButtons[row][col].setText("_");
          break;
        case Tic:
          gameButtons[row][col].setText("X");
          break;
        case Tac:
          gameButtons[row][col].setText("O");
          break;
      }
    }
  }
}
```

```java
public TTTMark getMark(int row, int col) {
  return this.field[row][col];
}

public boolean setMark(TTTMark mark, int row, int col) {
  TTTMark currentMark = getMark(row, col);
  if(currentMark != TTTMark.None) {
    return false;
  }else{
    this.field[row][col] = mark;
    if(this.currentPlayer == TTTMark.Tic) {
      this.currentPlayer = TTTMark.Tac;
    }else{
      this.currentPlayer = TTTMark.Tic;
    }
    return true;
  }
}
```

Feld besetzt?

Setzen und Spieler wechseln

```java
public void onClick(View view) {
  boolean success = false;
  int x = -1, y = -1;
  switch (view.getId()) {
    case R.id.ttb00:
      x = 0; y = 0;
      break;
    case R.id.ttb01:
      x = 0; y = 1;
      break;
     // fehlende einsetzen
  }
  success = this.gameModel.setMark(this.gameModel.getCurrentPlayer(), x, y);
  if(!success) {
    String text = "Das Feld war schon belegt!";
    Toast.makeText(getApplicationContext(), text, Toast.LENGTH_LONG).show();
  }
}
```

```java
public void onResetClick(View view) {
    this.gameModel.resetField();
    this.updateUiWithModel();
}
```

Was fehlt jetzt noch?

Ausständige Funktionalität

- Gewinner / Unentschieden ermitteln (betrifft `TicTacToeModel`)
- Meldungen an die Benutzer (betrifft `TicTacToeActivity`)

```
public TTTMark getWinner() {
  for(int row=0;row<GAME_SIZE;row++) {
    if(field[row][0] != TTTMark.None &&
       field[row][0] == field[row][1] && field[row][1] == field[row][2]) {
      return field[row][0];
    }
  }
  for(int col=0;col<GAME_SIZE;col++) {
    if(field[0][col] != TTTMark.None &&
       field[0][col] == field[1][col] && field[1][col] == field[2][col]) {
      return field[col][0];
    }
  }
  if(field[1][1] != TTTMark.None) {
    if(field[0][0] == field[1][1] && field[1][1] == field[2][2]) {
      return field[1][1];
    }
    if(field[2][0] == field[1][1] && field[1][1] == field[0][2]) {
      return field[1][1];
    }
  }
  return TTTMark.None;
}
```

```
public boolean isDraw() {
  if(getWinner() == TTTMark.None) {
    for(int row=0;row<GAME_SIZE;row++) {
      for(int col=0;col<GAME_SIZE;col++) {
        if(this.field[row][col] == TTTMark.None) {
          return false;
        }
      }
    }
    return true;
  }else {
    return false;
  }
}
```

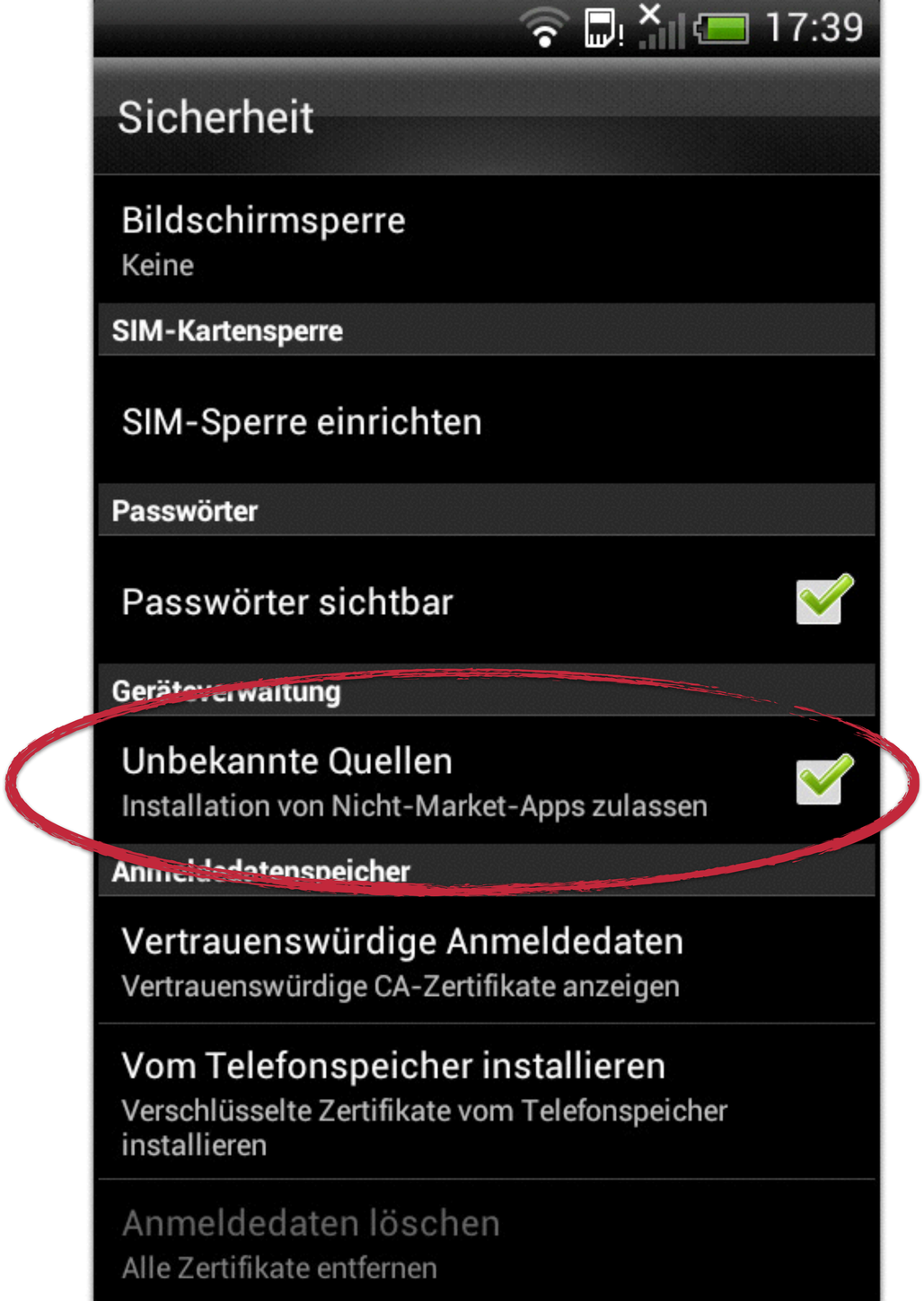
TicTacToeActivity.java

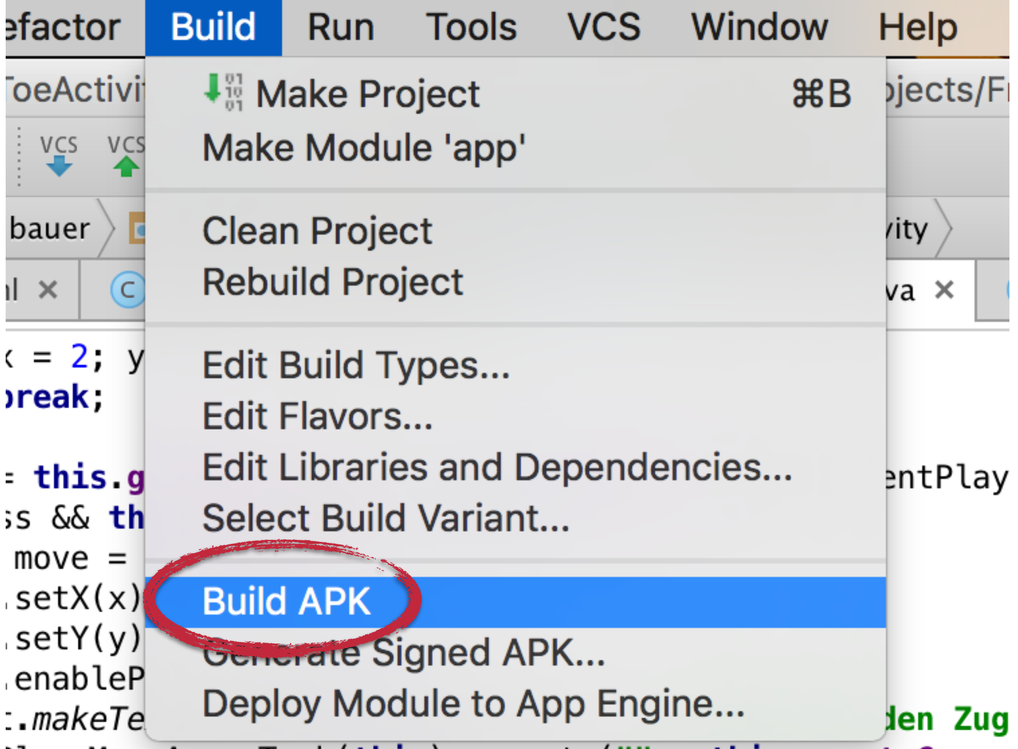
```
private void checkForWinner(boolean markResult) {
  if(markResult) {
    this.updateUiWithModel();
    TicTacToeModel.TTTMark winner = this.gameModel.getWinner();
    if(winner != TicTacToeModel.TTTMark.None) {
      String text = winner.toString() + " hat gewonnen!";
      Toast.makeText(getApplicationContext(), text, Toast.LENGTH_LONG).show();
    }else{
      if(this.gameModel.isDraw()) {
        Toast.makeText(getApplicationContext(), "Unentschieden!", Toast.LENGTH_LONG).show();
      }
    }
  }else{
    String text = "Das Feld war schon belegt!";
    Toast.makeText(getApplicationContext(), text, Toast.LENGTH_LONG).show();
  }
}
```

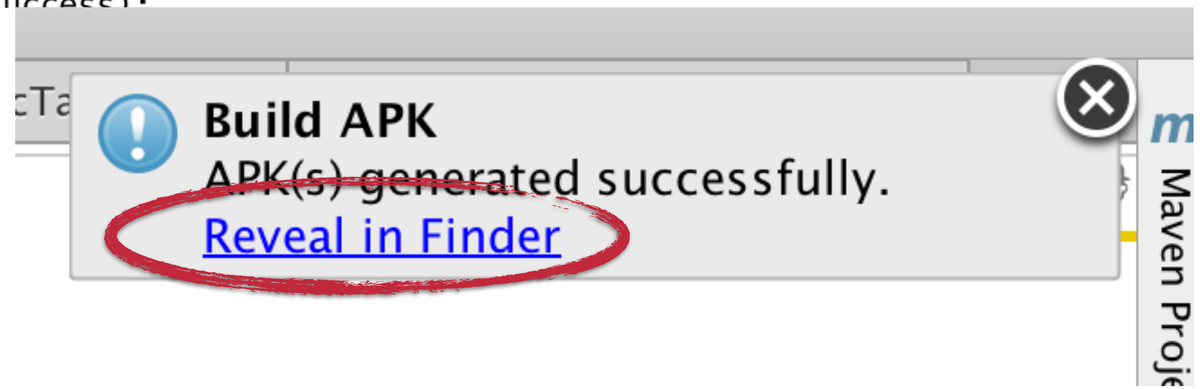
Methode **`onClick`** (letzte Zeile ersetzen)

```
checkForWinner(success);
```


Zeit das Spiel zu probieren!







APK per Dropbox oder e-Mail
aufs Handy und dort weiterspielen!

Hilfe ich bin viel zu schnell für diesen Workshop!

Multiplayer via Internet

- Das Institut für Telekooperation betreibt einen **REST TicTacToe Service**
- Öffentlich erreichbar unter: `http://www.tk.jku.at/tictactoe/`
- Über den Dienst kann ein Spiel (Gegner) angefragt werden und die Spieler können sich dann gegenseitig Moves schicken

Service URLs

- `http://www.tk.jku.at/tictactoe/games`
  kann per HTTP **POST** mit einem Benutzernamen dazu genutzt werden ein Spiel anzulegen; der Dienst antwortet mit einer **GameID** sobald **zwei Spieler** in einem **Spiel** sind

- `http://www.tk.jku.at/tictactoe/games/<gameid>`
  kann per GET angefragt werden um Details zum Spiel (Spielernamen) zu bekommen

Service URLs

- `http://www.tk.jku.at/tictactoe/games/<gameid>/nextMove`
  Kann per **GET** angefragt werden um den nächsten Zug vom Gegner abzurufen oder per **POST** mit einem Zug beschrieben werden

POST: Matthias

POST: Gabi

2837261

2837261

2837261, x=1, y=2

2837261, x=1, y=2

2837261, x=1, y=1

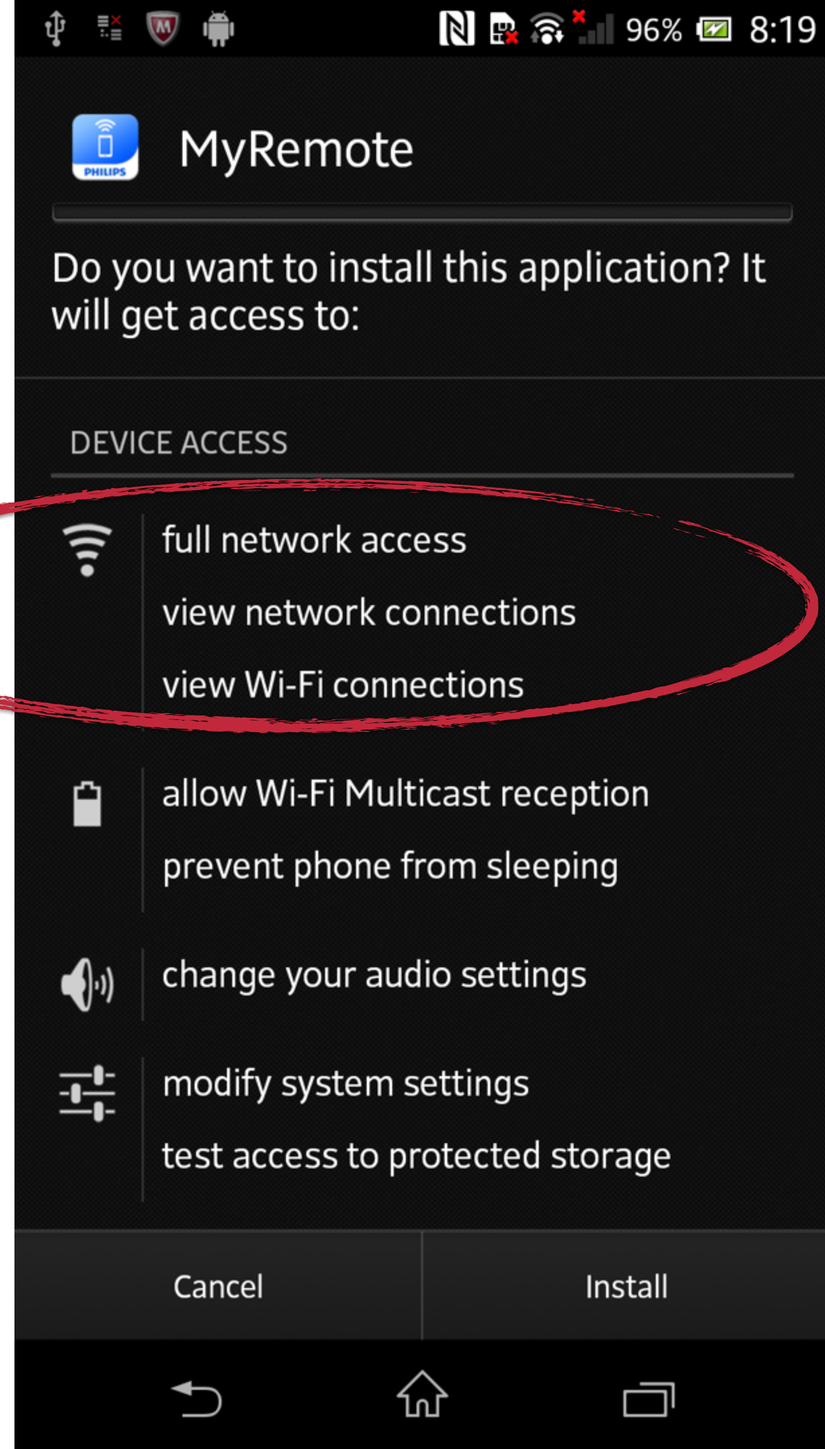
2837261, x=1, y=1

http://www.tk.jku.at/tictactoe/games

Netzwerkzugriff in Android erlauben

- In der Datei **`manifests/AndroidManifest.xml`** unter **`<manifest>`** eine Permission eintragen

```
<uses-permission
android:name="android.permission.INTERNET" />
```



REST API Retrofit einbinden

- In der Datei
  `build.gradle (Module: app)`
  folgende Abhängigkeiten eintragen
  dann synchronisieren

```
compile 'com.squareup.retrofit2:retrofit:2.0.0-beta4'
compile 'com.squareup.retrofit2:converter-gson:2.0.0-beta4'
```

Beans für die Datenhaltung

- `Player` enthält nur den Namen eines Spielers (`playerName`)
- `Game` enthält alle Details zu einem Spiel
  - Spieler
  - Game-ID
  - Anzahl der Spieler
  - Nächster Spielzug
- `Move` kann einen Spielzug darstellen

Player.java

```
public class Player {

    private String playerName;

    public String getPlayerName() {
        return playerName;
    }

    public void setPlayerName(String playerName) {
        this.playerName = playerName;
    }

}
```

Game.java

```
public class Game {

    private long id;

    private String[] playerNames;

    private int numberOfRegisteredPlayers;

    private Move nextMove;

    public Move getNextMove() {
        return nextMove;
    }

    public void setNextMove(Move nextMove) {
        this.nextMove = nextMove;
    }

    public long getId() {
        return id;
    }

    public void setId(long id) {
        this.id = id;
    }
```

```
  public String[] getPlayerNames() {
        return playerNames;
    }

    public void setPlayerNames(String[]
playerNames) {
        this.playerNames = playerNames;
    }

    public int getNumberOfRegisteredPlayers() {
        return numberOfRegisteredPlayers;
    }

    public void setNumberOfRegisteredPlayers(int
numberOfRegisteredPlayers) {
        this.numberOfRegisteredPlayers =
numberOfRegisteredPlayers;
    }

}
```

Move.java

```
public class Move {

    private int x, y;

    public int getX() { return x; }

    public void setX(int x) { this.x = x; }

    public int getY() { return y; }

    public void setY(int y) { this.y = y; }
}
```

TicTacToeService.java

```
public interface TicTacToeService {

    @POST("games")
    public Call<Game> createGame(@Body Player player);

    @POST("games/{gameId}/nextMove")
    public Call<Move> placeMove(@Body Move move, @Path("gameId") String gameId);

    @GET("games/{gameId}/nextMove")
    public Call<Move> awaitMove(@Path("gameId") String gameId);

}
```

ServiceFactory.java

```
public class ServiceFactory {

    public static TicTacToeService getService() {
        OkHttpClient httpClient = new OkHttpClient.Builder().
                readTimeout(180, TimeUnit.SECONDS).
                connectTimeout(60, TimeUnit.SECONDS).build();

        Retrofit retrofit = new Retrofit.Builder()
                .baseUrl("http://www.tk.jku.at/tictactoe/")
                .addConverterFactory(GsonConverterFactory.create())
                .build();

        TicTacToeService ticTacToeService =
                retrofit.create(TicTacToeService.class);
        return ticTacToeService;
    }
}
```

Prozesse in Android

- **Netzwerk-Kommunikation** darf nur in einem **Hintergrund**-Thread laufen > `AsyncTask` verwenden!
- Aktualisierung der **Benutzeroberfläche** darf nur im **UI-Thread** erfolgen

CreateGameAsyncTask.java

```
public class CreateGameAsyncTask extends AsyncTask<Player, Integer, Game> {

    private TicTacToeActivity connectedActivity;

    public CreateGameAsyncTask(TicTacToeActivity activity) {
        this.connectedActivity = activity;
    }

    @Override
    protected Game doInBackground(Player... params) {
        Call<Game> game = ServiceFactory.getService().createGame(params[0]);
        try {
            Game body = game.execute().body();
            this.connectedActivity.restGameResult(body);
            return body;
        } catch (IOException e) {
            e.printStackTrace();
        }
        return null;
    }
}
```

TicTacToeActivity.java

```
new CreateGameAsyncTask(this).execute(player);
Toast.makeText(getApplicationContext(),
    "Warte auf einen würdigen Gegner", Toast.LENGTH_LONG).show();
```

Neuen Button “Internet Spiel” einbauen

Handler Funktion erzeugen
obigen Code einbauen

Ergebnisse aus dem Netzwerk-Thread
werden über die **`restGameResult`** abgearbeitet

```
public void restGameResult(Game game) {
    if(game != null) {
        Log.e(“restGameResult",
            "Got a remote game " + game.getId());
```

Wie geht’s weiter?

- Das Projekt liegt auf GitHub unter:
  `https://github.com/steima/FroyoTicTacToe`

- Das Projekt kann direkt in AndroidStudio gecloned werden

JOHANNES KEPLER
UNIVERSITÄT LINZ

Matthias Steinbauer
matthias.steinbauer@jku.at
@climbinggeek