



# Mobile Technologies & Smart Devices

2020~2021

**overtref jezelf**



# Specifieke Resources

- ▶ Van veel resources kan je verschillende versies maken
- ▶ De specifieke versie zet je in een map met dezelfde naam als voor de standaard resource, gevolgd door een streepje (-) en dan een keyword dat aangeeft waar deze versie voor is
- ▶ Android zoekt zelf uit aan de hand van zijn instellingen of hij de specifieke versie gebruik, of terugvalt op de standaard resource.

Bijv.

/res/values/  
/res/values-nl/

layout  
layout-land

- ▶ Wat zou er in layout-land staan?

<https://developer.android.com/guide/topics/resources/providing-resources.html>

# Huiswerk

- ▶ Wat is het Manifest?

# Rechten / Security / Privacy

- ▶ Voor elke app maakt Android een gebruiker aan
- ▶ De rechten voor deze gebruiker staan in het Manifest
- ▶ De gebruiker van het systeem (de telefoon) geeft toestemming voor deze rechten
- ▶ Apps hebben geen directe toegang tot andere apps, maar communiceren via Android (Intent).  
Voor activities binnen dezelfde app doe je dit ook
- ▶ Apps hebben een eigen (afgeschermd) opslag voor gegevens

# Constanten

- ▶ Ziet er in code uit als een variabele, maar kan niet veranderd worden
  - ▶ Naamconventie: gebruik alleen hoofdletters
- 
- ▶ In java bestaan 'echte' constanten niet, maar maken we variabelen 'final static'

```
public final static String LOG_TAG = "Hello World";  
  
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    Log.d(LOG_TAG, "App created");
```

[https://en.wikipedia.org/wiki/Constant\\_\(computer\\_programming\)](https://en.wikipedia.org/wiki/Constant_(computer_programming))

# Constanten

- ▶ Voorkomt tikfouten
- ▶ Voorkomt gebruik niet toegestane waarden
- ▶ Vergroot de leesbaarheid van je code
- ▶ Vergroot beheersbaarheid van je code (waardes die meerdere keren gebruikt worden op één plek aan te passen)

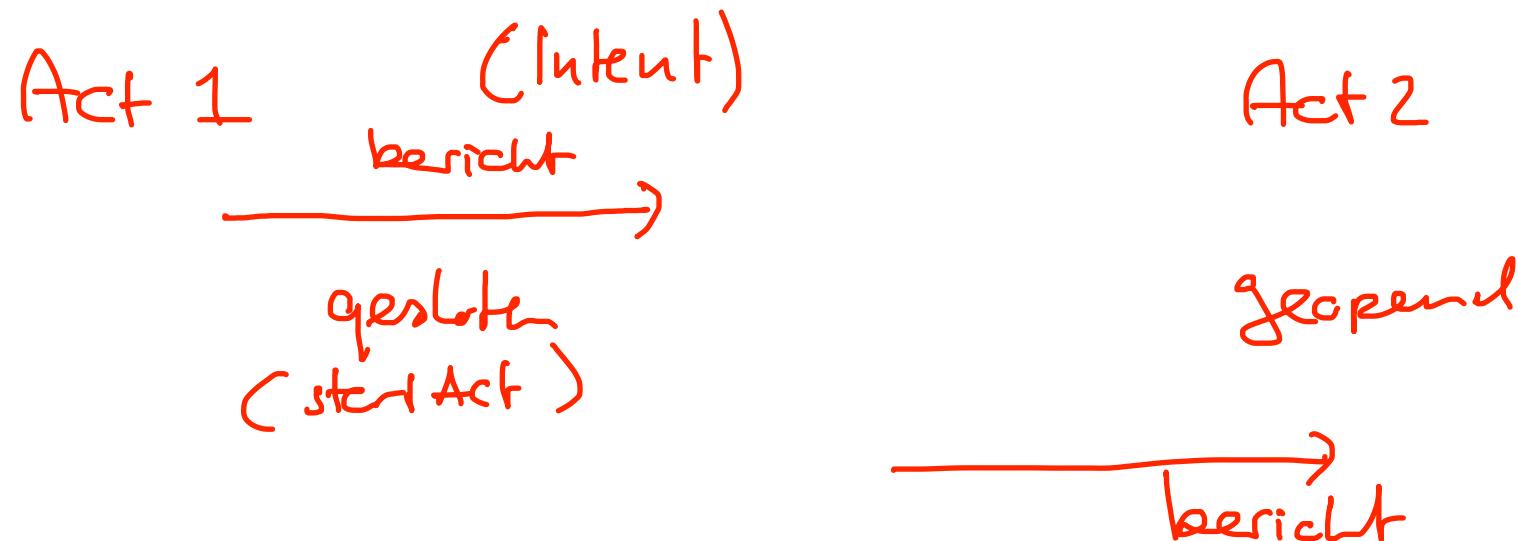
- ▶ Gebruik zo min mogelijk vaste waardes in je code

"Week 2" → LOG-TAG

15 → MAX\_SPEED

# Data doorgeven tussen activities

Android FL



# Data doorgeven tussen activities

- ▶ Geef informatie door via een intent  
*naar variabele*

```
Intent i = new Intent(this, OtherActivity.class);
i.putExtra("name", myName); ← waarde
startActivity (i);
```

```
// geopende activity intent uitlezen
Intent i = getIntent();
String myName = i.getStringExtra("name");
```

Act 1  
gesloten

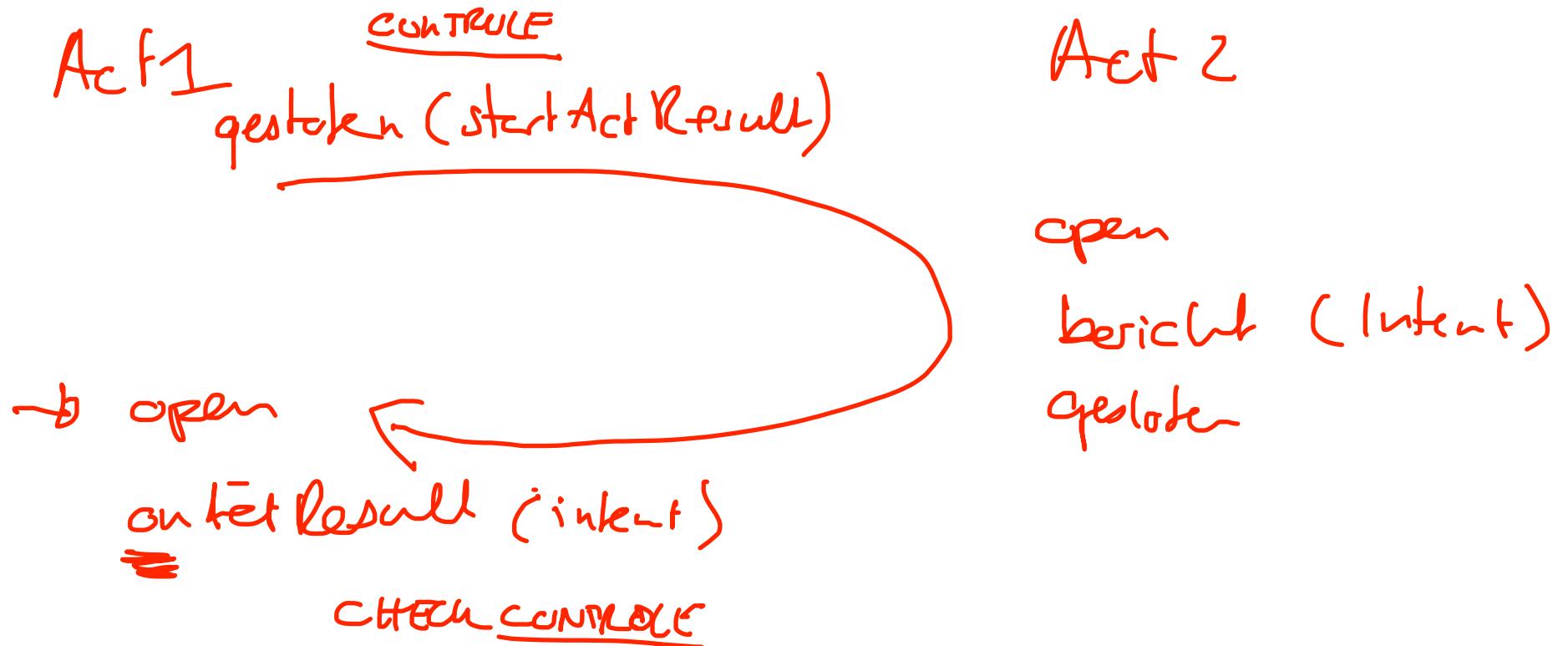
Act 2  
open

- ▶ Waarom is "name" in bovenstaand voorbeeld geen goede keuze?  
Wat kunnen we beter gebruiken?

↑  
Ondert te liever  
een constante te gebruiken

# Data doorgeven tussen activities

Android FL



# Data doorgeven tussen activities

- ▶ Open Activity for Result, en geef een code door voor controle
- ▶ Result wordt in intent gezet
- ▶ Bij terugkeer wordt onActivityResult geladen, check op RESULT\_OK en de juiste controle code voor je data uit de intent haalt

Act1  
Intent i = new Intent(this, ExplicitlyLoadedActivity.class);  
startActivityForResult(i, CONTROLE\_CODE);

← type van bericht (Android)  
onthaalt

Act2  
// geopende activity, geeft result terug  
Intent i = new Intent();  
i.putExtra(THETEXT, theText);  
setResult(RESULT\_OK, i);  
finish();

Act1  
// gebruik result  
@Override  
protected void onActivityResult(int requestCode, int resultCode, Intent data) {  
if ((resultCode == RESULT\_OK) && (requestCode == CONTROLE\_CODE))  
 String theText = data.getStringExtra(ExplicitlyLoadedActivity.THETEXT);  
}

# Data opslag

- ▶ Net gezien: Voor elke app maakt Android een aparte user aan
- ▶ XML-file in home-dir van de app-user
- ▶ Beveiliging automatisch geregeld door linux waarop Android gebouwd is

# Persistent Data (Single Activity)

- ▶ In user Prefs opslaan, niet zichtbaar voor gebruiker
- ▶ getPreferences(MODE\_PRIVATE)
- ▶ naamloos: voor enkele activity
- ▶ onPause
- ▶ onCreate, of handmatig

```
@Override  
protected void onPause() {  
    super.onPause();  
    | getPreferences(MODE_PRIVATE).edit().putString(PREF_REFERENCE,  
    |     stringToStore).commit();  
}  
    ↑ waarde  
  
// ophalen in onCreate  
stringToRetrieve = getPreferences(MODE_PRIVATE).  
    getString(PREF_REFERENCE, default);
```



# Persistent Data (Meerdere Activities)

- ▶ In user Prefs opslaan, niet zichtbaar voor gebruiker
- ▶ getSharedPreferences(PREF\_FILE\_NAME, MODE\_PRIVATE)
- ▶ Met naam, ook in andere activity (binnen app) te gebruiken
- ▶ onPause
- ▶ onCreate, of handmatig

```
@Override  
protected void onPause() {  
    super.onPause();  
    getSharedPreferences(PREF_FILE_NAME,  
        MODE_PRIVATE).edit().putString(PREF_REFERENCE, stringToStore).commit();  
}  
  
// ophalen in onCreate  
stringToRetrieve = getSharedPreferences(PREF_FILE_NAME, MODE_PRIVATE).  
    getString(PREF_REFERENCE, default);
```

# Persistent Data (Settings)

- ▶ Aanmaken gelijk aan gewone Activity
- ▶ PreferenceActivity uitbreiden
- ▶ Settings toevoegen (res/xml/settings.xml)
- ▶ Maak getters en setters voor de instellingen
- ▶ Opslaan is al voor je geregeld

```
// In onCreate in plaats van normale content laden  
addPreferencesFromResource(R.xml.settings);  
  
// waarden moeten exact overeen komen met waarden in settings.xml  
// gebruik van vaste waarden houdt het beter beheersbaar  
private static final String OPT_LOGO = "showLogo" ;  
private static final boolean OPT_LOGO_DEF = true;  
  
public static boolean getLogo(Context context) {  
    return  
        PreferenceManager.getDefaultSharedPreferences(context).getBoolean(OPT_LOGO,  
            OPT_LOGO_DEF);  
}
```

# Workshop

- ▶ Lees opdracht omschrijving en denk al na over je praktijkopdracht

Ko

# Afbeeldingsbronnen

- <https://mobilebcitcomputing.wordpress.com/2015/05/23/hello-world/>
- <https://www.vectorstock.com/royalty-free-vector/medical-face-mask-flat-vector-28176994>
- [https://commons.wikimedia.org/wiki/File:Android\\_robot.svg](https://commons.wikimedia.org/wiki/File:Android_robot.svg)