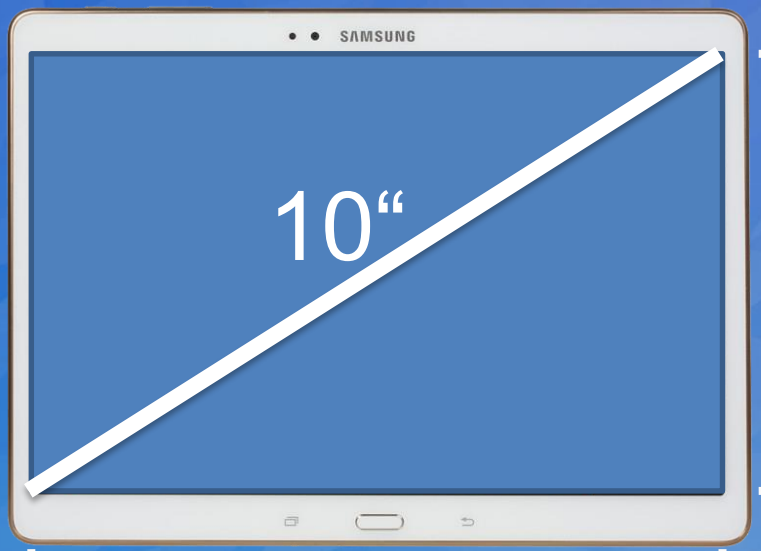
How to Design an Android App

1 App for all Android Devices

Auflösung und Pixeldichte



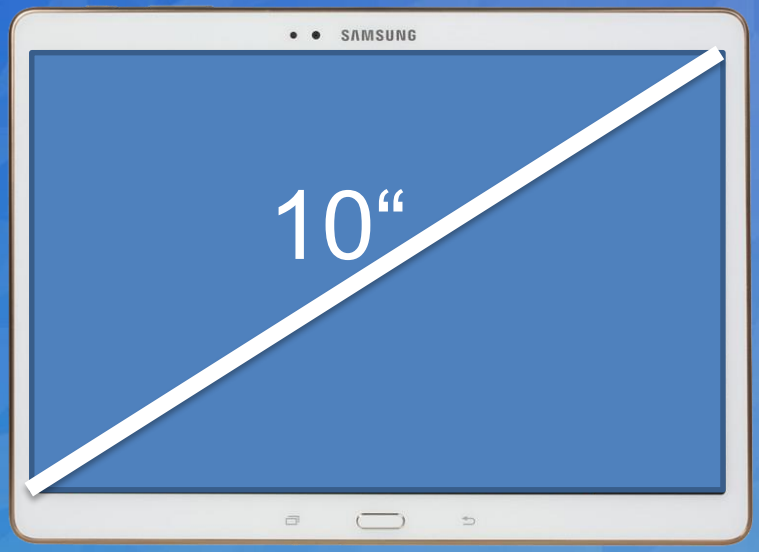
1080px

1920px

1920px

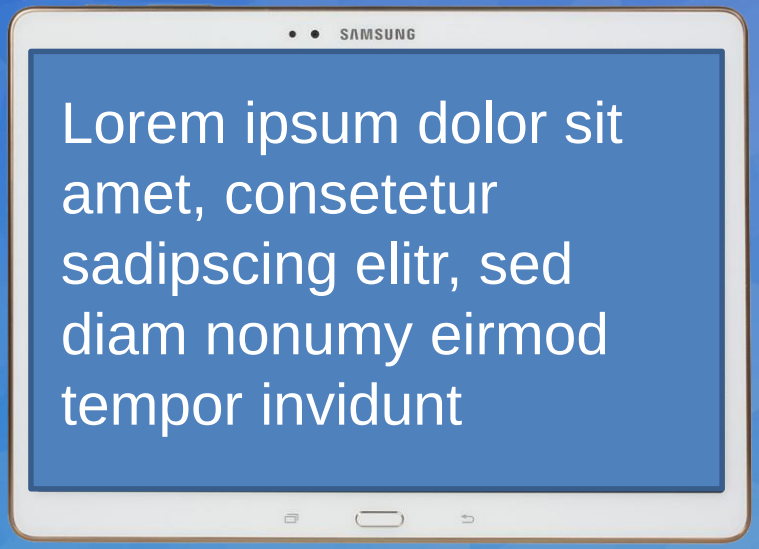
1080px

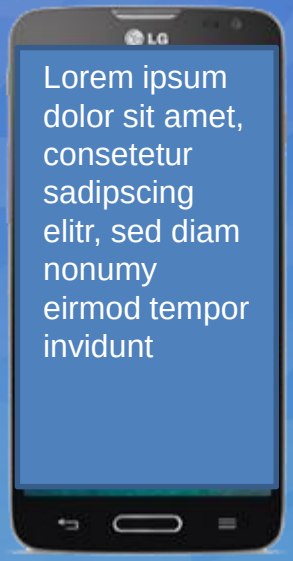
Auflösung und Pixeldichte



- Gleiche Aufösung
- Unterschiedliche Pixeldichten

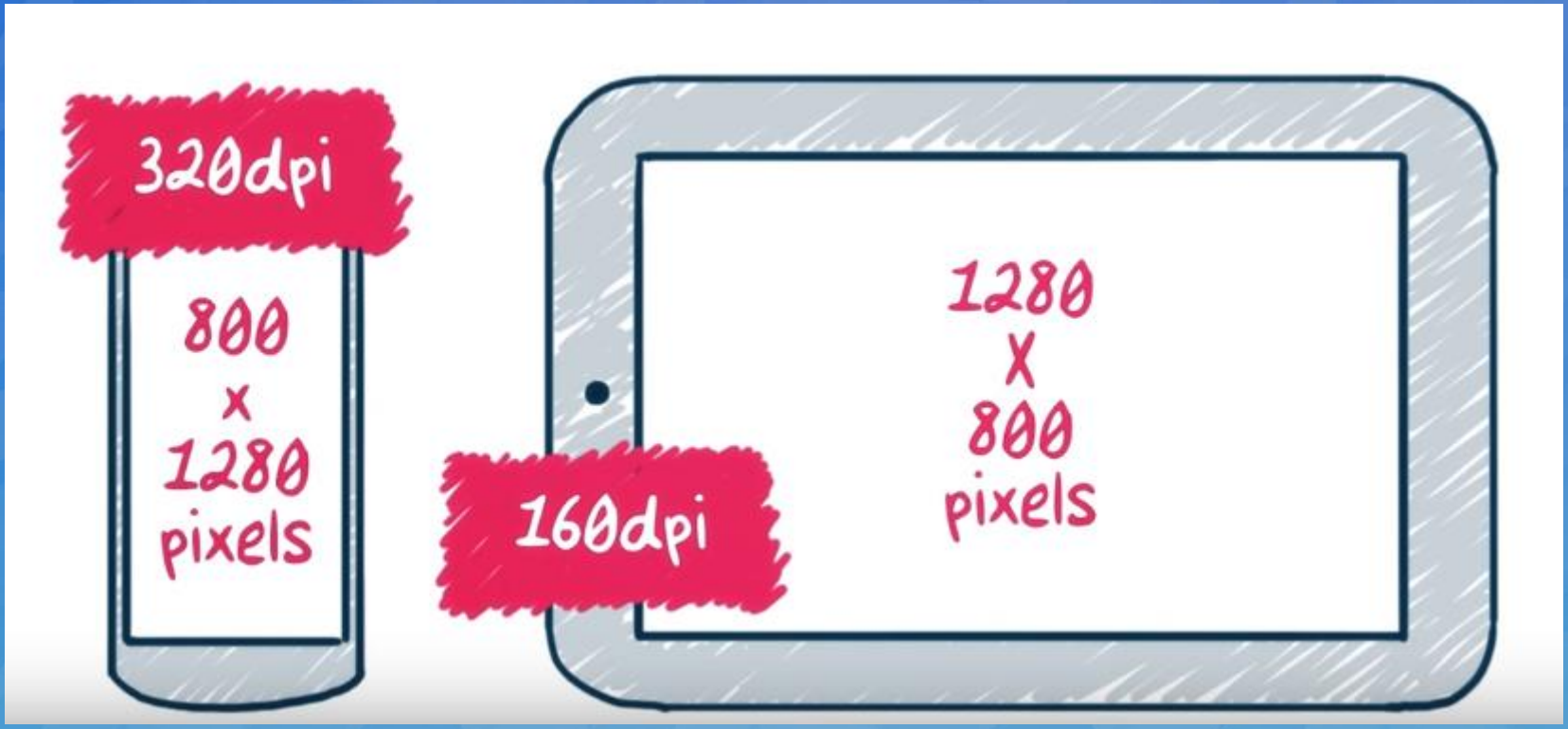
Auflösung und Pixeldichte



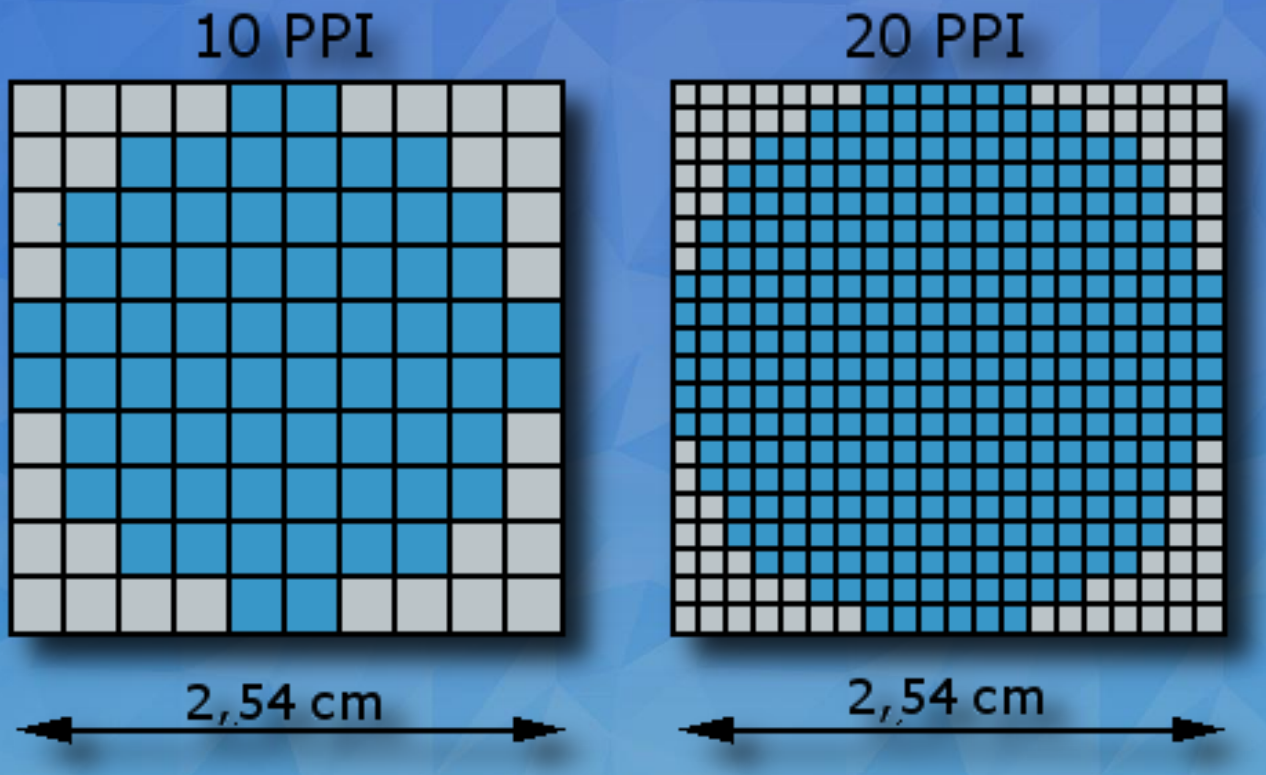


- Mögliche Probleme:
  - Zu kleine / große Texte, Buttons, etc.

Auflösung und Pixeldichte



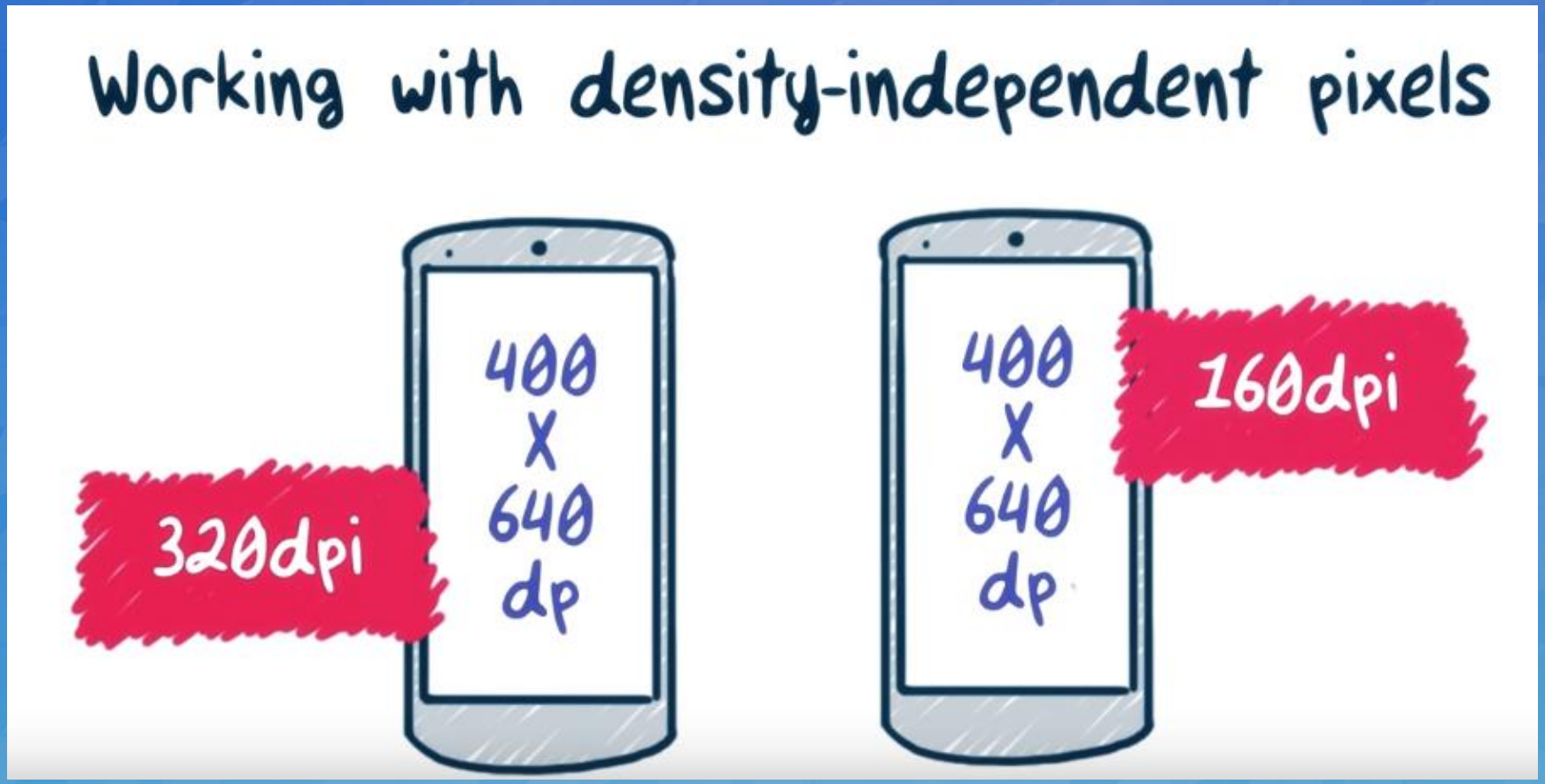
DPI



Density indenpendent Pixels

- Android bietet das System der „Density independent pixels“, mit der Einheit „dp“.
- Mit dp muss sich der Entwickler nicht um Auflösungen und Pixeldichten kümmern, sondern kann für die tatsächliche Bildschirm-Größe designen.

Density indenpendent Pixels



Layouts

- Layouts definieren u.a. Größe und Anordnung von Buttons, Listen, Textfeldern, etc.

```
<Button
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/button_text"
    ... />
```

XML

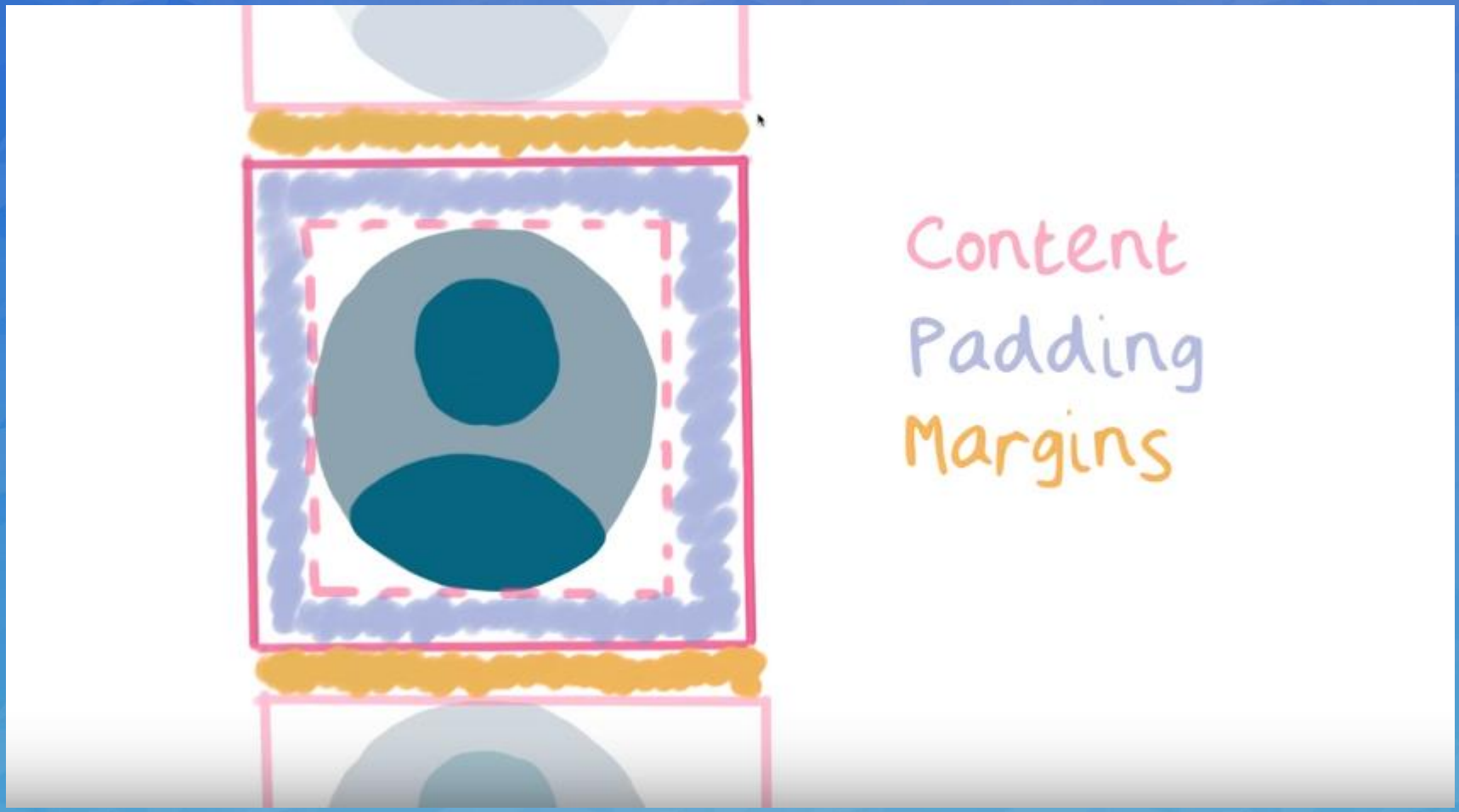
```
Button myButton = new Button(this);
myButton.setText("Push Me");

LinearLayout ll =
(LinearLayout)findViewById(R.id.buttonlay
out);
LayoutParams lp = new
LayoutParams(LayoutParams.MATCH_PARENT,
T, LayoutParams.WRAP_CONTENT);

 ll.addView(myButton, lp);
```

Java Code

Content, Padding & Margins



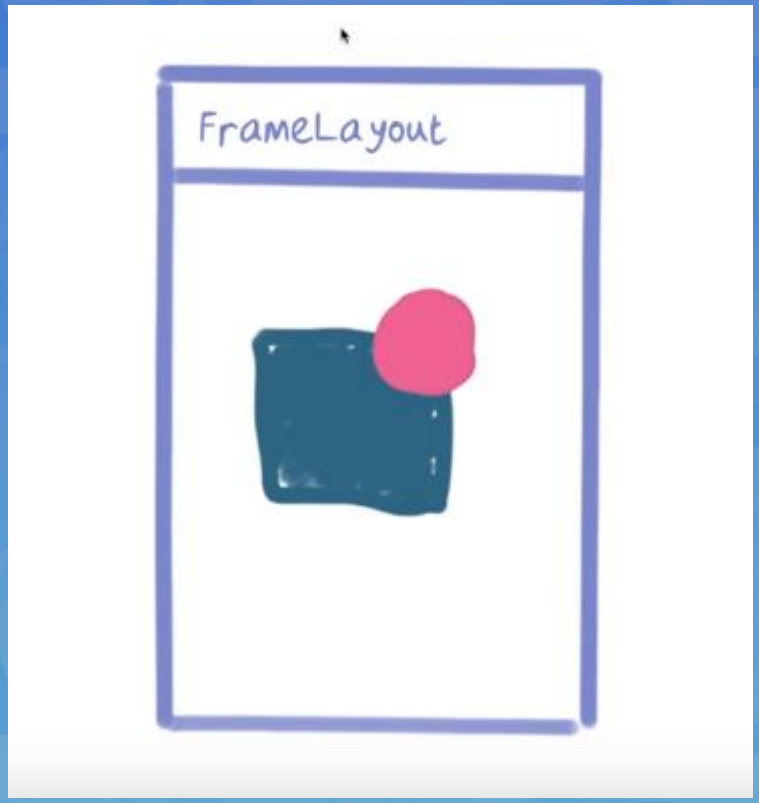
Types of Layouts

- Frame Layout
- Linear Layout
- Grid Layout
- Relative Layout

- ScrollView
- ListView
- ViewPager

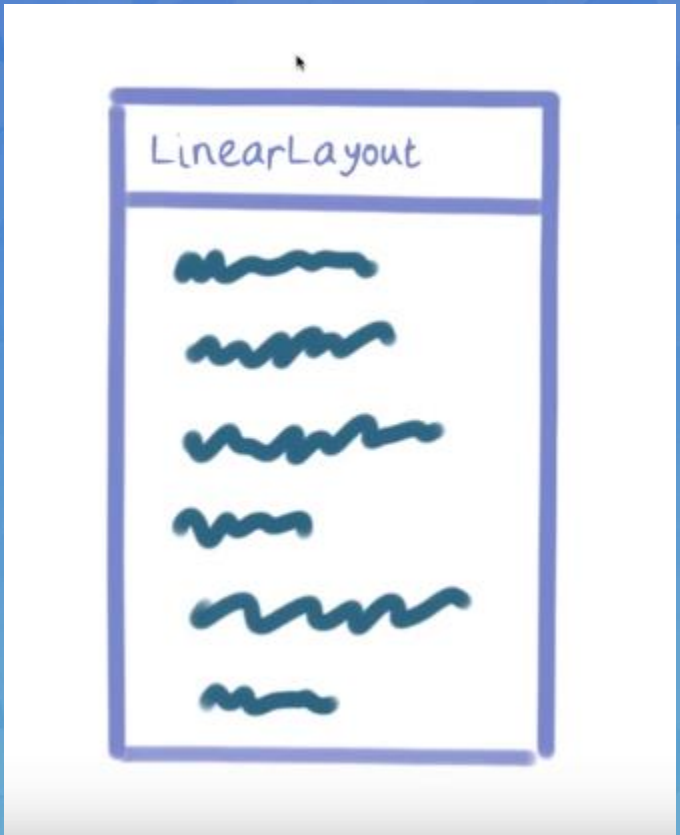
Frame Layout

- Nützlich für überlappende Objekte



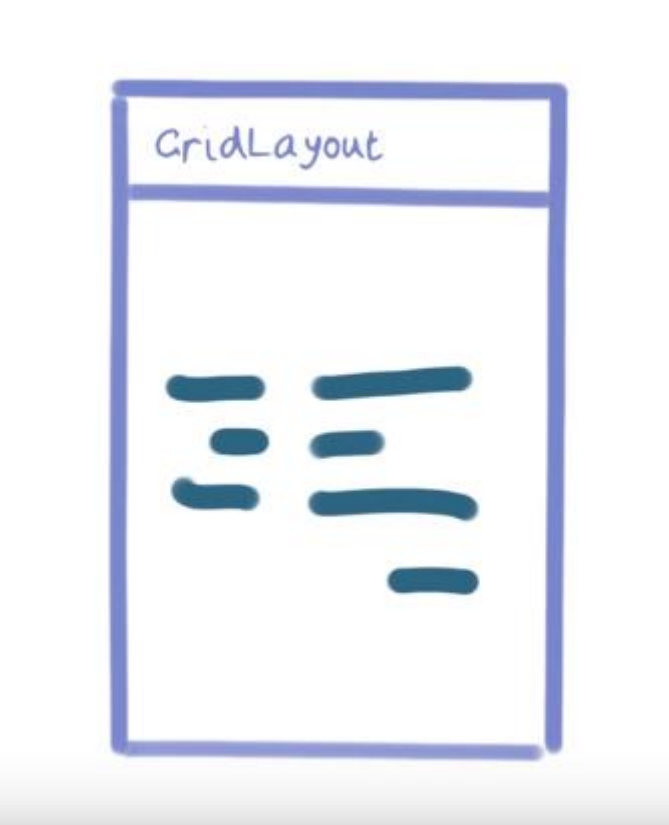
Linear Layout

- Ordnet Elemente vertikal/horizontal



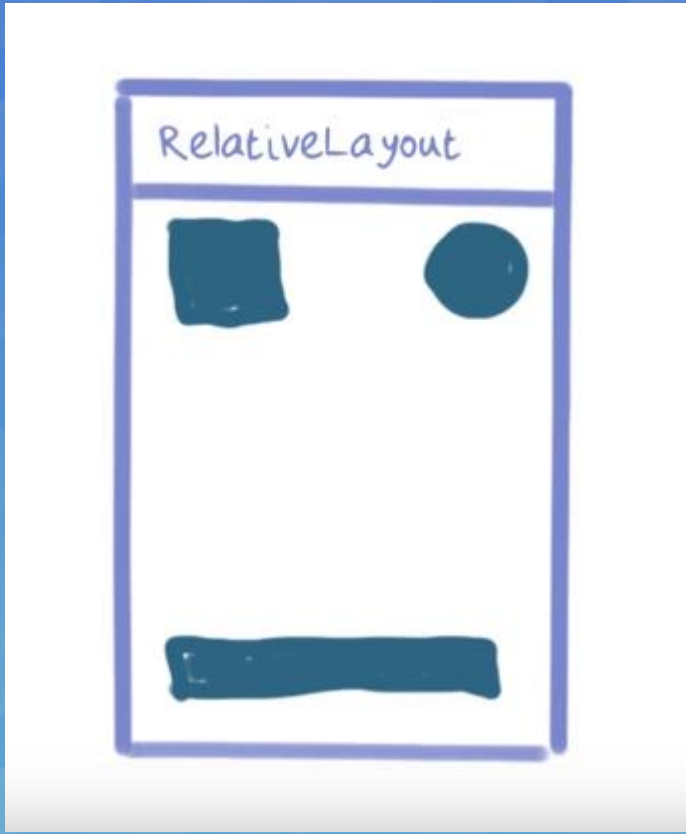
Grid Layout

- Tabellen-ähnliche Anordnung



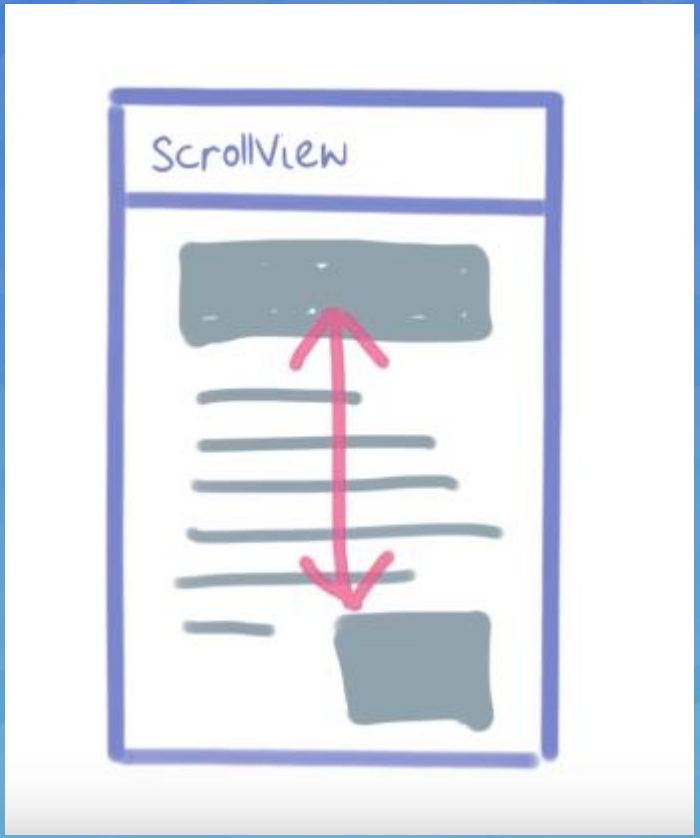
Relative Layout

- Ordnet Elemente relativ zueinander an.

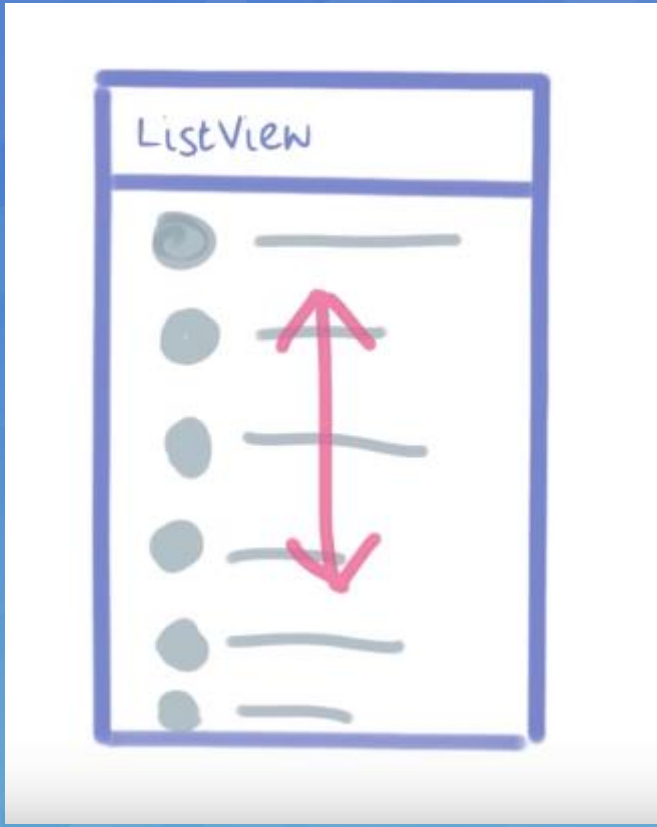


Scroll View

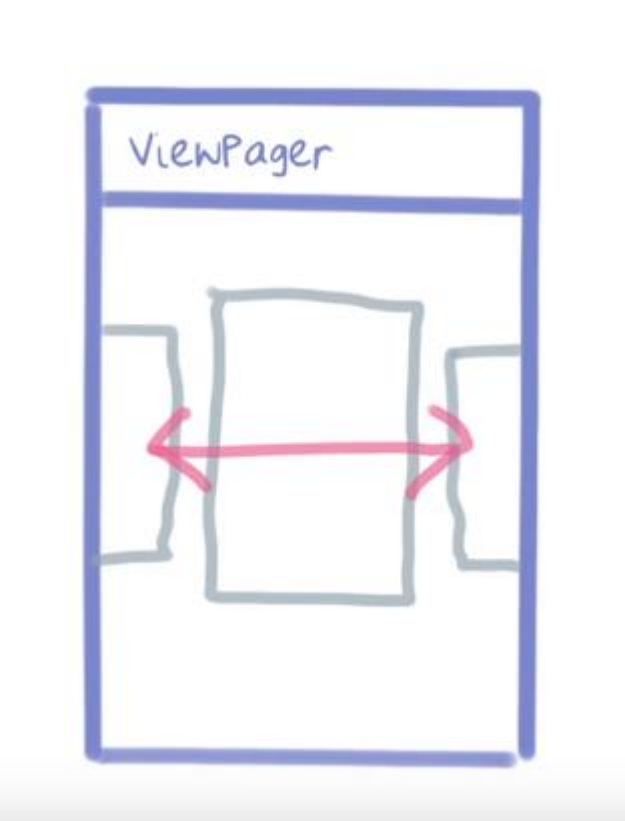
- Hält nur ein Element, das bewegt werden kann.



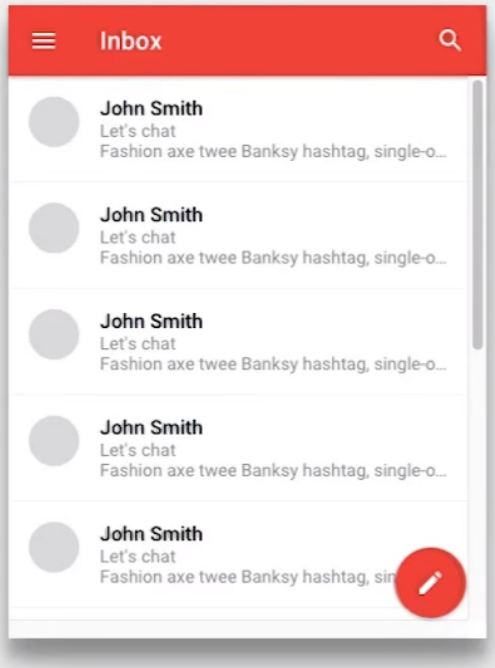
List View



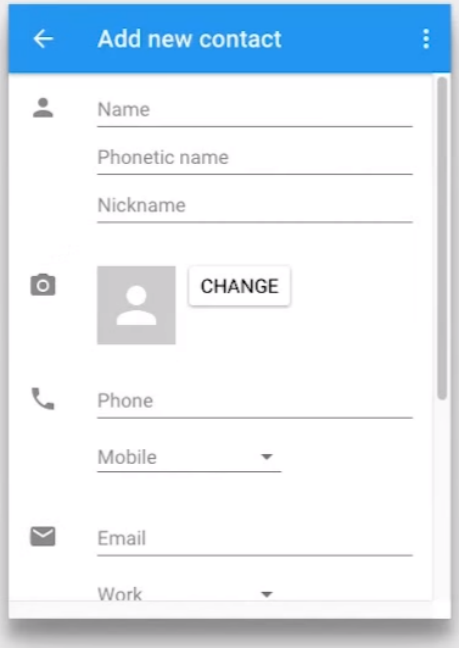
View Pager



Layout Example: Mail-App

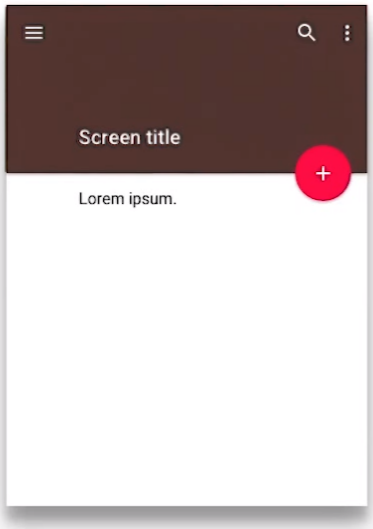


Layout Example: Add Contact



Layout Example: App Store

Layout Quiz



A

- FrameLayout
  - TextView
    layout_marginTop: 168dp
    text: @string/lorem_ipsum
  - Toolbar
    layout_height: 168dp
  - Button
    layout_gravity: right
    layout_marginTop: 140dp

B

- LinearLayout
  orientation: vertical
  - Toolbar
    layout_height: 168dp
  - Button
    layout_gravity: right
    layout_marginTop: 140dp
  - TextView
    layout_marginTop: 168dp
    text: @string/lorem_ipsum

C

- FrameLayout
  - TextView
    layout_marginTop: 168dp
    text: @string/lorem_ipsum
  - Button
    layout_gravity: right
    layout_marginTop: 140dp
  - Toolbar
    layout_height: 168dp

D

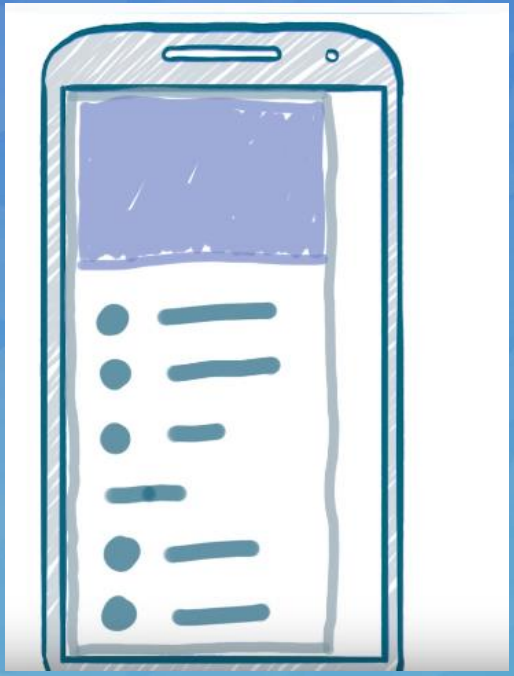
- FrameLayout
  - LinearLayout
    orientation: vertical
    - Toolbar
      layout_height: 168dp
    - TextView
      layout_weight: 1
      text: @string/lorem_ipsum
  - Button
    layout_gravity: right
    layout_marginTop: 140dp

E All of the above

Common UI Design Patterns

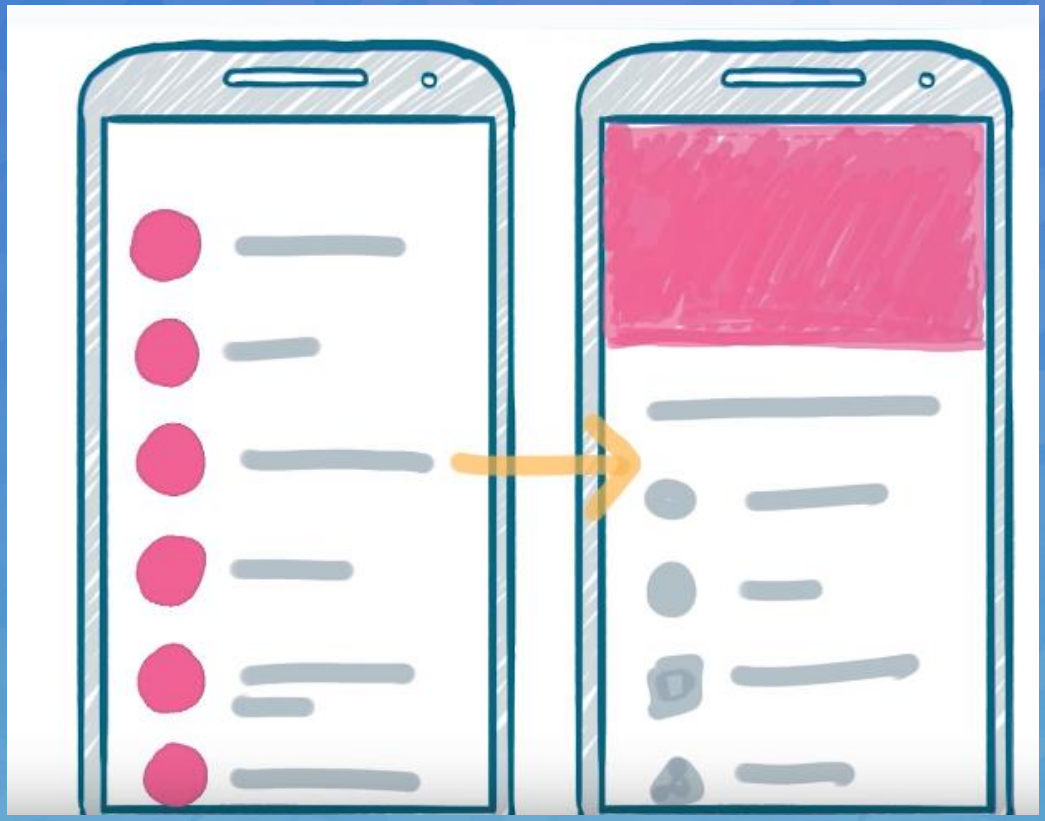
- Toolbar

- Navigation Drawer



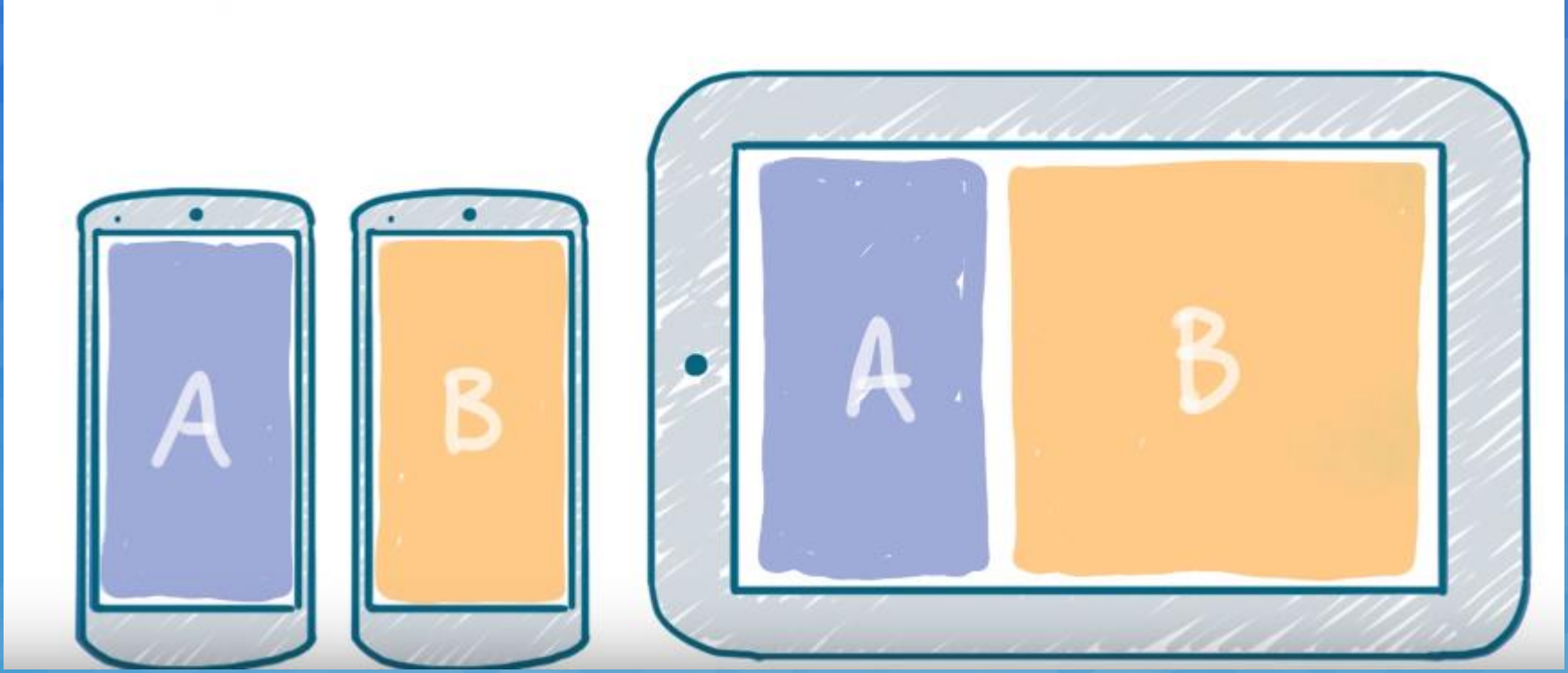
Common UI Design Patterns

- List to details



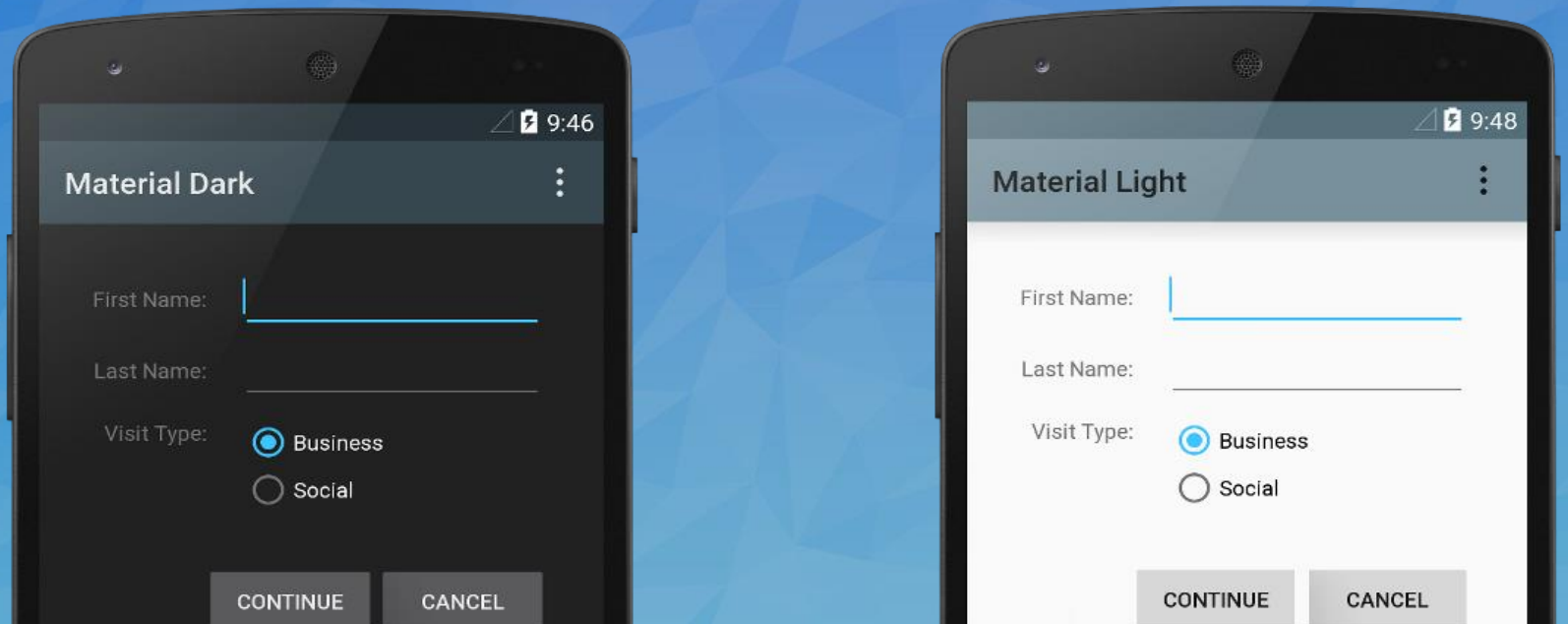
Common UI Design Patterns

- Multipane

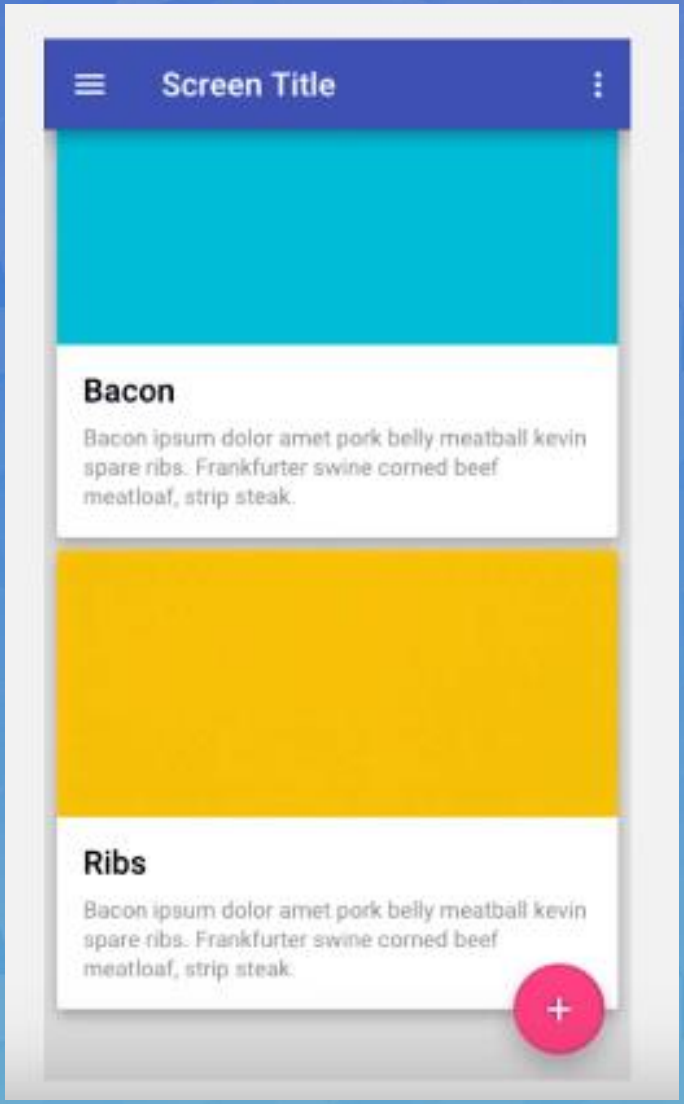


Material Design

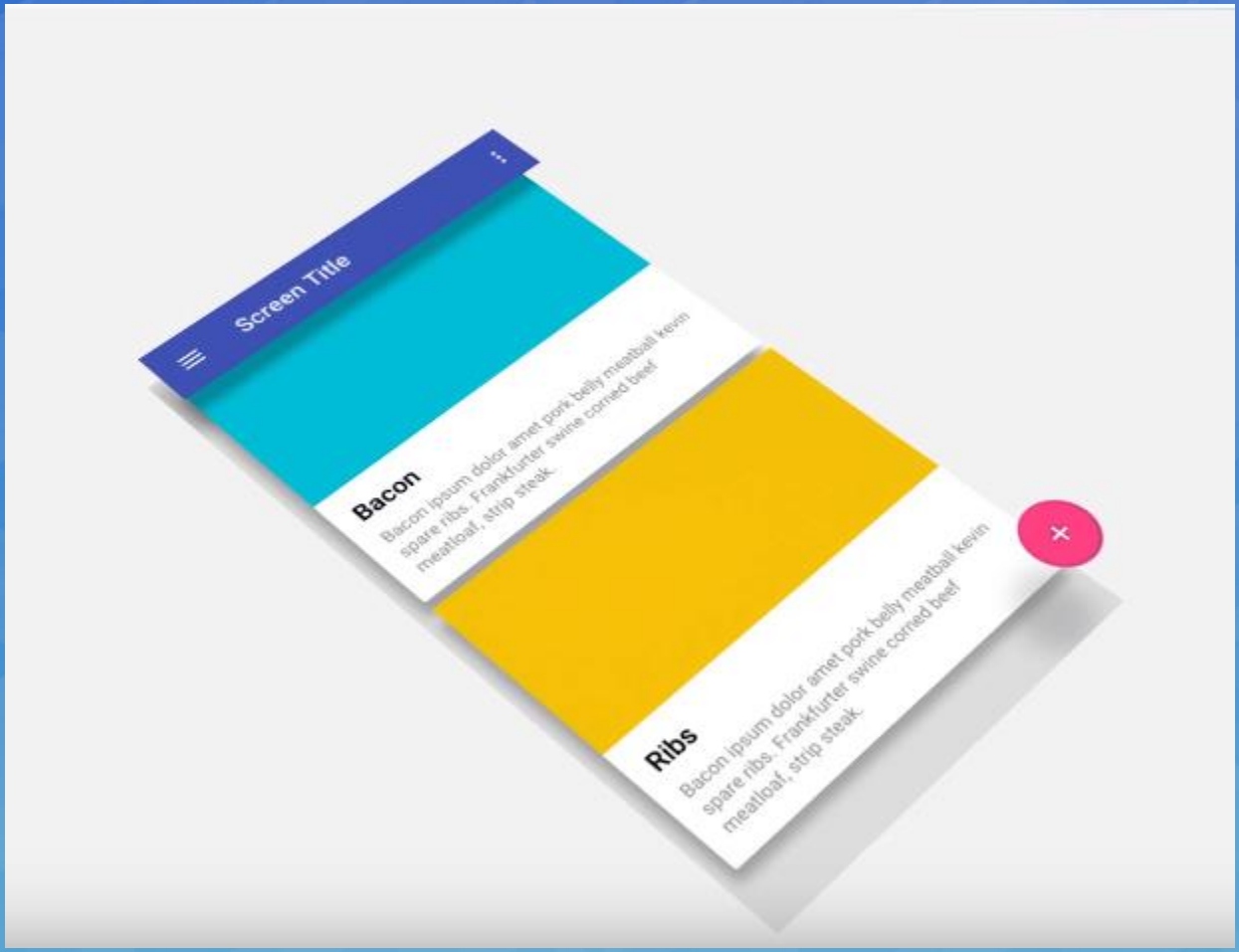
- = neue UI-Elemente und Richtlinien um eine moderne, benutzerfreundliche Android-App zu erstellen.
- Seit Android 5.0



Material Design: Surfaces

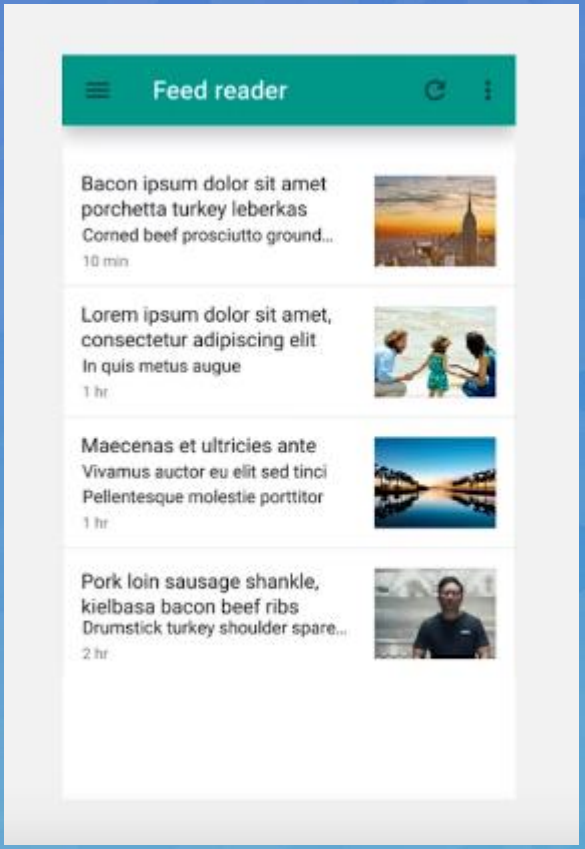


Material Design: Surfaces



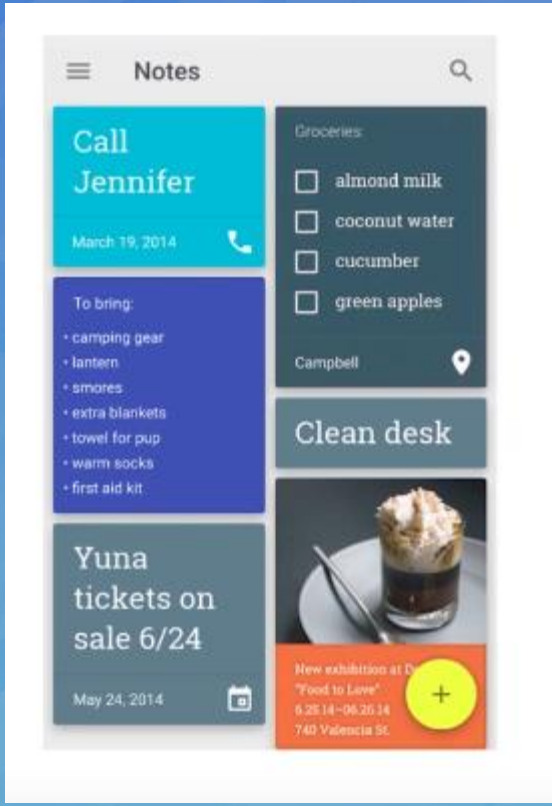
Surfaces

- Gleichartige Elemente auf einem Surface anordnern:



Surfaces

- Unterschiedliche Elemente aufeigenen Surfaces anordnen:



Gestalt laws

Gestalt laws - Similarity

similarity

anomally

Gestalt laws - Continuation

Gestalt laws - Closure

Gestalt laws - Proximity

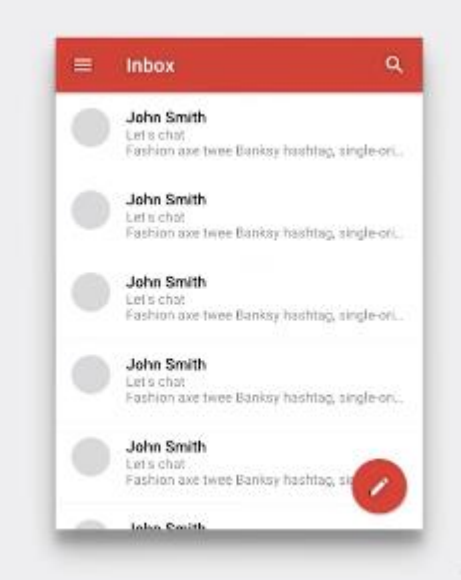
Gestalt laws – Common fate

Adaptive Design

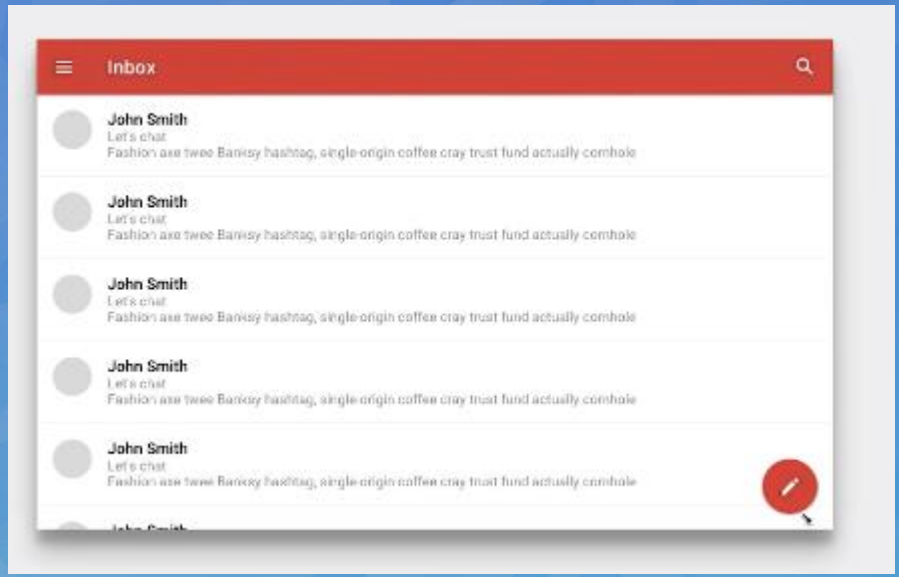
- Herausforderung:
  - Unterschiedliche Bildschirmgrößen und -ausrichtungen erfordern unterschiedliche User Interfaces.

Adaptive Design: Beispiel

Mail-App, übersichtlich auf einem Smartphone



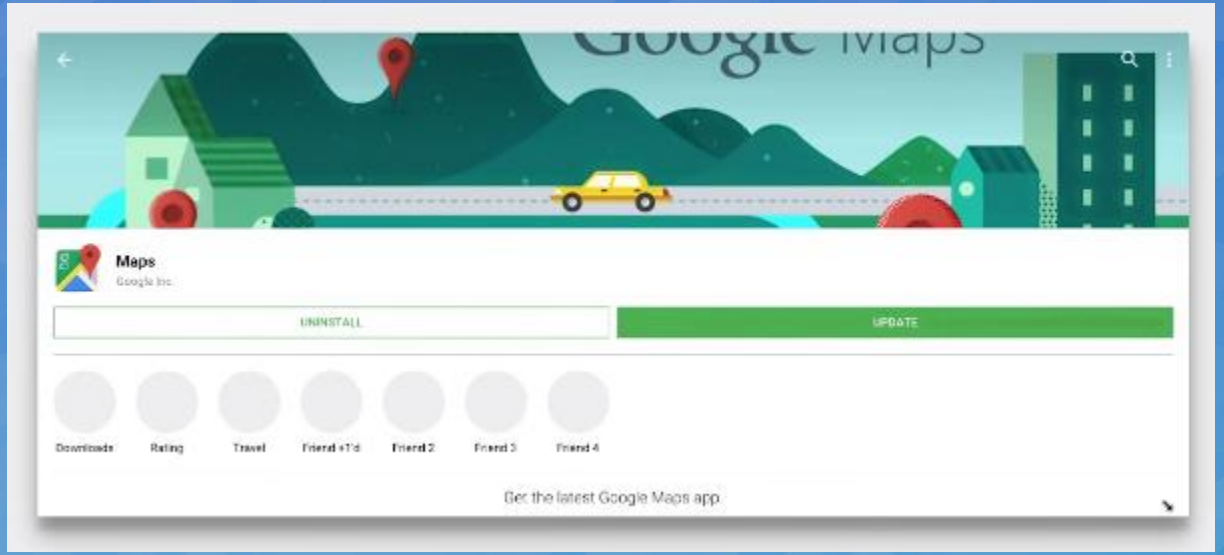
Ungenutzter Platz auf einem Tablet, Textfelder schlechter lesbar.



Adaptive Design: Beispiel

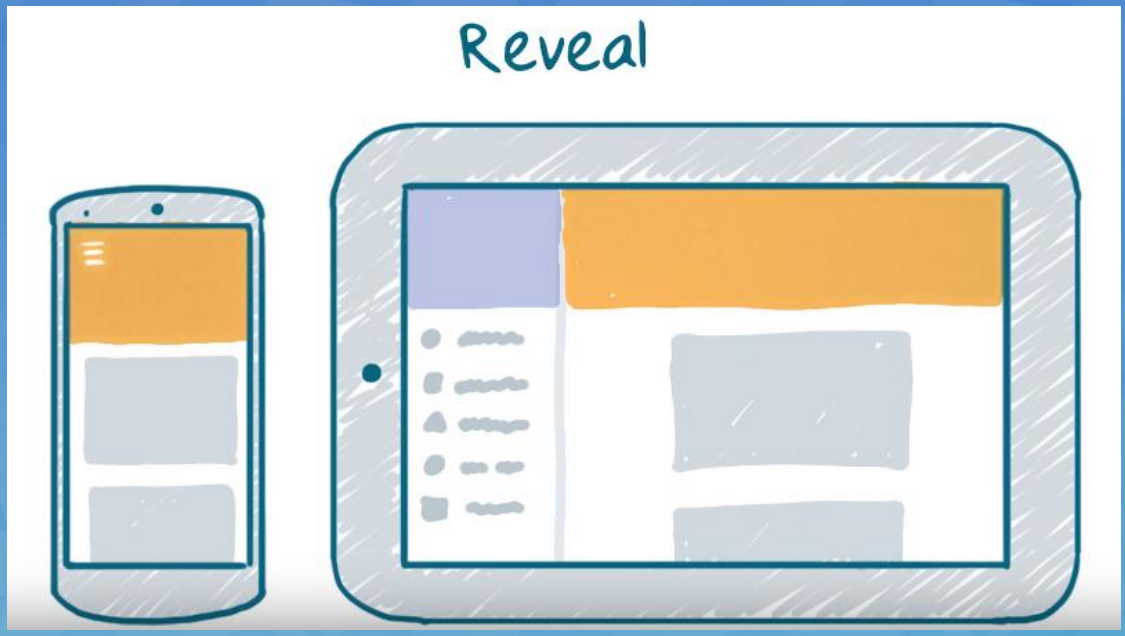
Play-Store, übersichtlich auf einem Smartphone

Ungenutzter Platz auf einem Tablet, Bild nur teilweise sichtbar, Buttons extrem breit

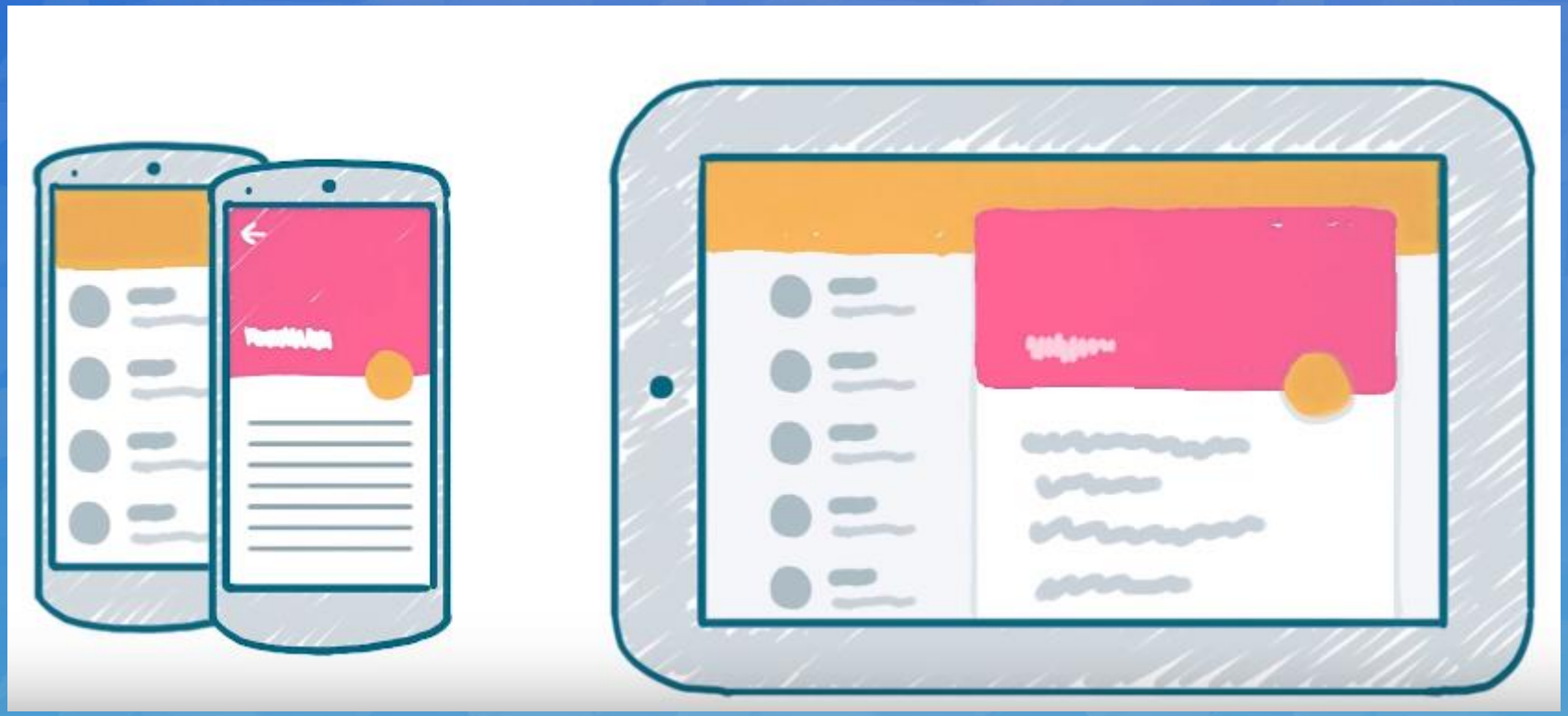


Adaptive Design: Lösungen

- „Reveal“ Prinzip: Zeige auf größerem Bildschirm UI-Elemente direkt an, die auf einem kleineren versteckt sind (zB Menü).



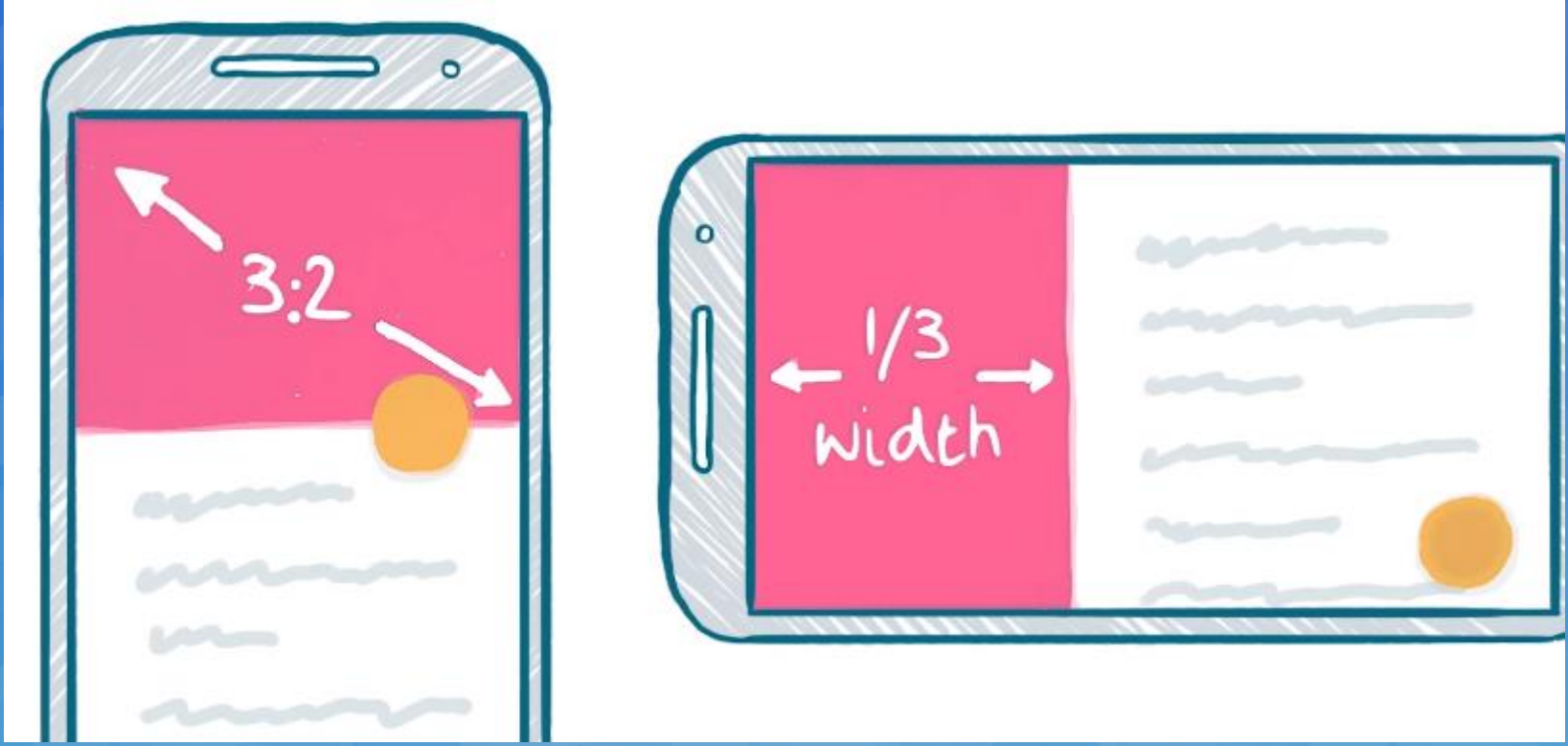
Adaptive Design: Lösungen



Adaptive Design: Lösungen

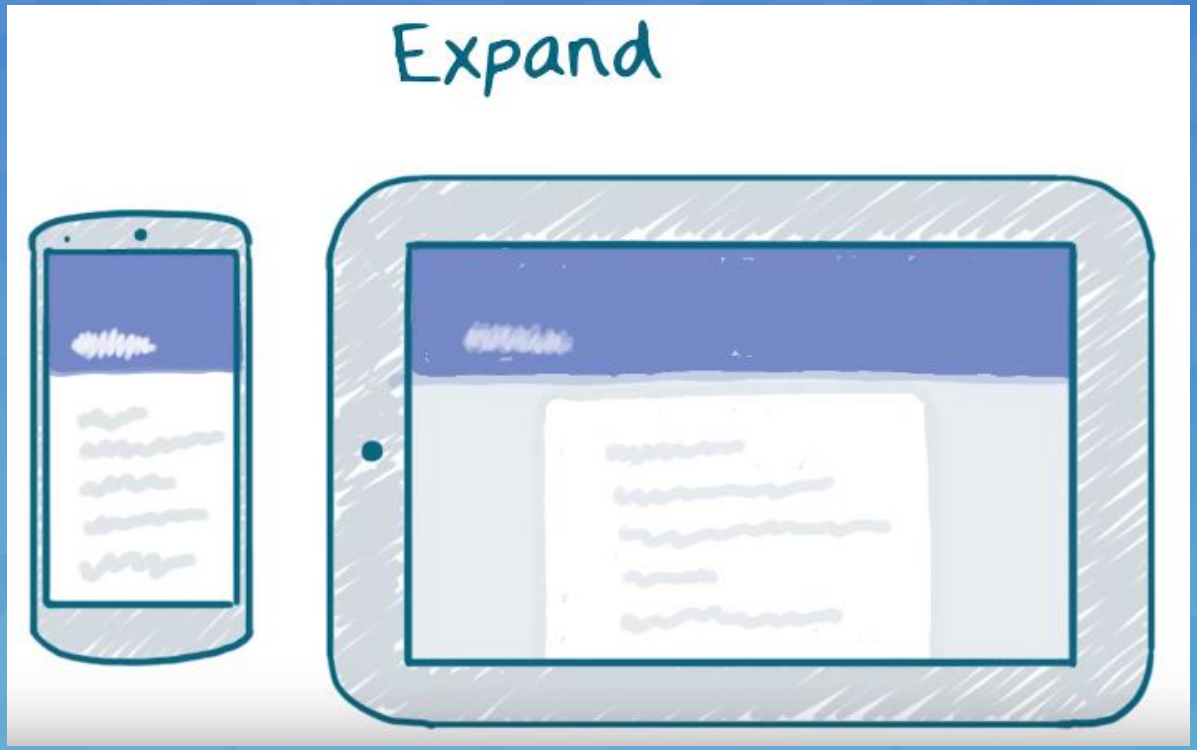
- „Reflow“ Prinzip: Unterschiedliche Anordnung und Größe von Elementen, je nach Bildschirmgröße.

Adaptive Design: Lösungen



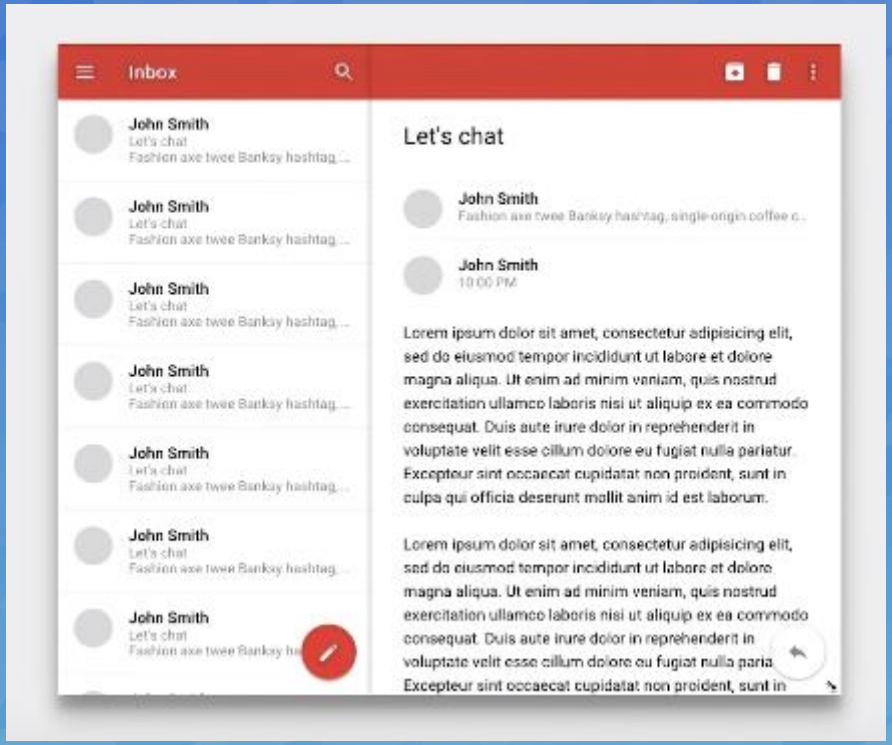
Adaptive Design: Lösungen

- „Expand“ Prinzip: Lasse bestimmte Elemente bis zu einem gewissen Punkt größer werden.



Adaptive Design

- Mail-App Lösung:



Adaptive Design

- Play-Store Lösung: