1 Designrichtlinien

Designrichtlinien sind ein wichtiges Element aller Applikationen bei denen der kommerzielle Erfolg stark vom Userinterface und dem visuellen Auftritt abhängt. Das dies bei unseren Zielsetzungen der Fall ist, ist es uns ein großes Anliegen, dass wir uns als Team im Vorhinein über das geplante Erscheinungsbild einig sind. Wir verwenden dabei Tipps und Designinstruktionen von Google, welche auf der Internetseite https://www.google.com/design/spec/material-design/introduction.html nachzulesen sind.

1.1 Der Splashscreen

Der Splashscreen ist der erste Screen den der User sieht. Hier wird meist das Logo dargestellt, während im Hintergrund die Daten geladen werden. Der Splashscreen sollte nicht zu lange sichtbar oder zu langweilig sein. Der Splashscreen kann folgendermaßen umgesetzt werden:

- ImageView: Mithilfe einer ImageView lässt sich ein einzelnes Bild darstellen und dieses Bild ist auch während der Laufzeit einfach austauschbar.
- VideoView: Ähnlich der ImageView lässt sich mithilfe einer VideoView ein einzelnes Video darstellen und man hat die Möglichkeit das Video zu starten oder zu pausieren.
- ProgressBar: Mithilfe einer ProgressBar ist es möglich Fortschritt grafisch darzustellen. Dabei ist auch möglich eine Cycling Animation anzuzeigen bei der dem User kein Fortschritt angezeigt wird, abgesehen davon dass die App etwas berechnet.

Der Splashscreen beinhaltet bei der FIPLY-App eine normale in der Mitte des Bildschirms zentrierte ImageView mit dem FIPLY-Logo, welche Präsenz 3500ms andauert wird. Der Hintergrund dieses Splashscreens ist weiß.

1.2 Animationen und Transactions

Eine Transition ist der Übergange zwischen zwei Activities. Der Übergang muss flüssig verlaufen und darf die User Experience nicht unterbrechen. Der User soll darauf aufmerksam gemacht werden wohin er seine Aufmerksamkeit richten sollte. Dies wird durch persistierende Objekte und präzise Bewegungen in den Übergängen erzielt. Diese Möglichkeiten stehen zur Verfügung:

- ViewSwitcher: Ein ViewSwitchers ist ein ViewAnimator mit dem man zwischen exakt 2 Views hin und herwechseln kann.
- TextSwitcher: Ein TextSwitcher ist ein ViewSwitcher der nur Text-Views auswechseln kann. Dieser wird verwendet um Labels zu animieren.
- ImageSwitcher: Ein ImageSwitcher ist ein ViewSwitcher der nur Bilder auswechseln kann. Dieser wird verwendet um das Wechseln von Bildern zu animieren.
- ViewFlipper: Ein ViewFlipper ist ein ViewAnimator mit dem man den Wechsel zwischen 2 oder mehr Views animieren kann.
- Fragment: Mithilfe von Fragments kann man mehrere Ebenen von Views innerhalb einer einzelnen Activity darstellen. Somit bildet ein Fragment ein auswechselbares Objekt, dass in anderen Activitys wiederverwendet werden kann.

Beschreibung unserer Umsetzung:

- Enter Transition: Die Enter-Transition besteht daraus, dass die aufgerufene Activity aus dem Button der Activity hervorgeht, welche sie ausgelöst hat. Dabei werden vergrößern sich jeweils die Eckpunkte eines Buttons und schmiegen sich der Form des Screens an, wobei die neue Activity gleichzeit durch einen transparenten Fade-In-Effekt erscheint. Es soll so wirken, als ob die geöffnete Activity aus dem Button hervorgeht. Wenn eine neue Activity nicht durch einen Button ausgelöst wird, wird diese einfach von der rechten Seite "eingeschoben".
- Exit Transition: Die Exit-Transition wird genau so wie die Enter-Transition durchgeführt werden, nur umgekehrt. Wenn die Activity geschlossen werden soll, verschwindet sie mit einem transparenten Fade-Out Effekt und die Eckpunkte der View schmiegen sich zurück an den Button wodurch sie ausgelöst wurde, sodass die andere Activity wieder erscheint. Falls eine Activity nicht durch einen Button ausgelöst wurde und sie geschlossen werden soll, wird sie nach rechts aus dem Screen "hinausgeschoben".
- Splashscreen Exit Transition: Der Splashscreen wird nach seiner Anzeigedauer durch einen transparenten Fade-Out Effekt verschwinden, sodass das Hauptmenü angezeigt wird.

1.3 Navigation

Als Navigation wird das Wechseln von einer Funktion in der App zur anderen bezeichnet. Dies wird meist durch Buttons erreicht und grafisch mit Transitions dargestellt. Die Buttons sollen auch ohne Text sprechend sein was sie bewirken bzw. wohin man mit ihnen navigiert. Dies kann durch bestimmte Zeichen auf den Buttons erreicht werden, z.B.: ein Zahnrad für die Einstellungen oder eine Lupe für eine Suchfunktion. Es sollen auch andere Methoden zur Navigation wie beispielsweise über den Bildschirm wischen verwendet werden. So lässt sich dies Umsetzen:

- Fragments: Mithilfe von Fragments kann man Navigation vortäuschen. Man wechselt dabei nicht zwischen den Activitys hin und her, sondern wechselt Teile der aktuellen Activitys aus um eine Navigation unnötig zu machen beziehungsweise vorzutäuschen.
- PopupMenu: Ein PopupMenu ist ein Menü das in einer View verankert ist und vor allem für Aktionen innerhalb einer Activity verwendet wird.
- ContextMenu: Ein ContextMenu ist ein Menü das vorallem Aktionen für ein spezielles Item bereitstellt. Ein ContextMenu wird vor allem bei ListViews oder GridViews eingesetzt um Aktionen wie Edit, Share oder Delete für einzelne Items bereitzustellen.
- Buttons: Ist ein Element einer View, das man entweder anklickt oder gedrückt hält, um Aktionen, z.B. Navigation, auszuführen.
- NavigationDrawer: Ist ein Element, dass in der linken Hälfte des Bildschirms angezeigt werden kann. Dies wird mittels einem DrawerLayout realisiert das eine ListView mit den Zielen der Navigation und ein anderes Layout mit dem Inhalt der Activity enthält.
- OptionsMenu: Ein OptionsMenu ist ein Menü das vor allem Aktionen für die aktuelle Activity bereitstellt. Bei älteren Geräten (API level 10 or niedriger) wird dieses OptionsMenu im unteren Teil des Bildschirms angezeigt. Ab API level 11 werden die Elemente des OptionMenus in der AppBar zur Verfügung gestellt.
- Zurücktaste am Gerät: Mithilfe der Zurücktaste, die in allen Androidgeräten verbaut ist, kann man auf die zuletzt besuchte Activity zurückspringen.

In der Applikation wird dies mittels der Technologie des OptionsMenu/AppBar und Buttons umgesetzt, um die Navigation der Applikation möglichst intuitiv für die Benutzer zu gestalten.

- NavigationDrawer: In dem NavigationDrawer werden Navigationen zu den Activities wie "Über uns" und "Profil bearbeiten" zur Verfügung gestellt, dessen Ansicht man in der Hauptansicht der Applikation auslösen kann. Ziel ist es, diesen NavigationDrawer auch innerhalb anderer Activities bereitzustellen.
- Buttons: Weiter Navigationen werden mittels Buttons ausgelöst:
  - Wechseln zwischen Hauptbildschirm und Trainingssessionanzeige.
  - Wechseln zwischen Hauptbildschirm und dem Menü zum generieren eines Trainingsplans.
  - Wechseln zwischen Hauptbildschirm und dem Statistikmenü.
  - Wechseln zwischen Hauptbildschirm und Trainingskatalogsmenü.
- Zurücktaste am Gerät: Mithilfe dieses Button wieder zur vorhergehenden Activity zurückspringen.

1.4 Formulare

Formulare sind der Ort an dem der User Daten in die App eingibt, wie beispielsweise beim Anlegen eines Userprofils. Formulare sollten so intuitiv wie möglich sein und so wenig wie möglich reine Texteingabe sein, wie z.B.: einen Slider anstatt eines Textfeldes für die Körpergröße. Manchmal ist jedoch eine Texteingabe unumgänglich, wie beispielsweise für die Namenseingabe. Behandlung mit Android Studio: (Necessary Evil-Steuerelemente)

- TextView: Eine TextView ist ein kompletter Texteditor, der in seiner Basisform allerdings kein bearbeiten des Textes erlaubt. Die Einsatzmöglichkeiten von TextViews sind beinahe unbegrenzt da Features wie Rechtschreibprüfung, Klickbare Links, Passwortfelder oder auch Eingabemethoden unterstützt werden.
- EditText: Ein EditText ist eine standardmäßig editierbare TextView.
- AutoComplete TextView: Mithilfe eines DropDownMenus werden dem User Vorschläge für die Textvervollständigung angezeigt die mittels Array definiert sind.
- Button: Ist ein Element einer View, das man entweder anklickt oder gedrückt hält, um Aktionen auszuführen.
- DatePicker: Mithilfe eines Datepickers kann man ein bestimmtes Datum über einen Spinner oder eine CalendarView auswählen.

- TimePicker: Mithilfe eines TimePickers kann man die Zeit auswählen.
- NumberPicker: Ein NumberPicker erlaubt dem User eine Zahl aus einer vordefinierten Zahlenmenge auszuwählen. Dazu werden 2 Buttons zur Verfügung gestellt die jeweils nach oben oder nach unten zählen.
- Switch: Ein Switch hat 2 Zustände (z.B: on and off) und der Zustand kann durch Klicken oder Ziehen geändert werden.
- CheckBox: Eine CheckBox hat 2 Zustände (checked and unchecked) und dieser kann durch Klicken geändert werden.
- RadioButton: Ein RadioButton hat 2 Zustände (checked and unchecked) und dieser kann durch Klicken geändert werden. Anders als bei einer RadioButtons werden meist in einer RadioGroup verwendet. Selektieren eines RadioButtons deselektiert alle anderen RadioButtons dieser Gruppe.
- ToggleButton: Ein ToggleButton hat 2 Zustände (z.B.: on and off) und der Zustand kann durch Klicken geändert werden. Der Zustand wird durch ein Aufleuchten des Buttons angezeigt.
- SeekBar: Eine SeekBar ist eine ProgressBar mit ziehbarem Slider um den Fortschritt zu setzen.
- RatingBar: Eine RatingBar ist eine SeekBar, die den Fortschritt in Sternen anzeigt. Der Fortschritt kann durch Klicken oder Ziehen verändert werden.
- Spinner: Ein Spinner ermöglicht dem User ein Kindelement aus einer Liste von Kindelementen auszuwählen. Dies wird mittels einem DropDownMenu realisiert.

Um den Benutzern eine komfortable Eingabe anzubieten, kommen überall dort intuitive Steuerelemente zum Einsatz, wo es möglich ist.

1.5 Datenausgabe

Datenausgabe ist der Teil der App wo dem User Information ausgegeben wird, wie beispielsweise der Übungskatalog. Die Datenausgabe soll so effizient wie möglich erfolgen. Suchfunktionen bzw. Filter sollen bei jeder Datenausgabe vorhanden sein. Dises Optionen stehen uns zur Auswahl:

- ProgressBar: Mithilfe einer ProgressBar ist es möglich Fortschritt grafisch darzustellen.
- RatingBar: Eine RatingBar ist eine SeekBar, die den Fortschritt in Sternen anzeigt. Optional kann der Fortschritt durch Klicken oder Ziehen verändert werden.
- ImageView: Mithilfe einer ImageView lässt sich ein einzelnes Bild darstellen und dieses Bild ist auch während der Laufzeit einfach austauschbar.
- VideoView/YoutubeAPI: Ähnlich der ImageView lässt sich mithilfe einer VideoView ein einzelnes Video darstellen und man hat die Möglichkeit das Video zu starten oder zu pausieren.
- MediaController: Ein MediaController ist eine View mit Steuerungselementen für den MediaPlayer. Der Controller sorgt dafür, dass die App synchron mit dem MediaPlayer läuft.
- CalendarView: Mithilfe einer CalendarView kann man ein Datum in einem Kalender darstellen oder auswählen.
- Clocks: Mithilfe einer TextClock kann man die aktuelle Zeit als formatierten String sowohl im 12-hour als auch im 24-hour Format anzeigen. Mithilfe einer AnalogClock kann man eine Uhrzeit an einer analogen Uhr anzeigen.
- StackView: Mithilfe einer StackView kann man immer ein Kindelement im Vordergrund anzeigen und sieht die restlichen Kindelemente dahinter angeordnet. Das angezeigte Element kann dabei durch jedes andere Kindelement ausgetauscht werden.
- ListView: Zeigt eine vertikale Liste von Elementen an.
- ExpandableListView: Zeigt eine vertikale Liste von Elementen an. Bei Klicken kann ein Element aufgeklappt werden um mehr Informationen anzuzeigen.
- WebView: Eine WebView kann Webseiten anzeigen ohne in den Webbrowser des Geräts wechseln zu müssen. In der WebView sind auch diverse Steuerelemente zur Navigation oder Textsuche enthalten.

Das ProgressBar Element kommt bei dem Anzeigen der Statistik zum Einsatz. Somit bekommen die User ein visuelles Feedback wie weit sie von ihrem

Ziel noch entfernt sind. Die VideoView/Youtube API wird beim Menüpunkt des Anzeigens der Details einer Übung verwendet. Jede Übung besitzt ein Vorschauvideo, welches durch dieses Element wiedergegeben wird. Die ExpandableListView ist die Anzeige des Übungskatalogs. Dieses Element beinhaltet alle zur Verfügung stehenden Übungen, welche durch Einstellungen gefiltert bzw. durchsucht werden können.

1.6 Style

Die gesamte Applikation bekommt harte Farben um Festigkeit zu symbolisieren. Die Hintergrundgfarbe aller Activities ist ein starkes Grau, die Entscheidung viel schlussendlich auf die Farbe mit dem Farbcode #424242. Alle Steuerelemente innerhalb der Applikation erhalten an den Eckpunkten eine 2px dünnen Konturradius mit unserer Hauptfarbe der Applikation, #009688. Um innerhalb der Elemente dem Farbthema treu zu bleiben, wie es von Google empfohlen wird, haben wir uns für #E0F2F1 entschieden. Diese Farbe wird als Hintergrundfarbe bei den Steuerelementen eingesetzt. Sie repräsentiert eine weiche Neutralität und übt dadurch keinen starken Kontrast mit den anderen Farben aus, wodurch den Usern ein visuell-angenehmes Farbenspektrum geboten wird.