# Klassen Planung (df)

## Klassen Planung für Activitys

Es sollen für jede Activity fünf Klassen implementiert werden, um die einzelnen Bereiche einer Activity getrennt behandeln zu können.

Zunächst soll es eine **Init** Klasse geben, welche von der AppCompactActivity Klasse abgeleitet sein soll, weswegen ihr Konstruktor beim Starten einer Activity automatisch aufgerufen werden wird. In diesem besagten Konstruktor sollen lediglich die anderen vier Klassen instanziiert werden.

Als erstes soll die **Gui** Klasse Instanziiert werden, welche zuerst einmal, der Programmatisch existierenden Acticity, das entsprechende Activity Layout zuweist. Die Klasse wird programmatischen Variablen enthalten , die den Komponenten der Activity entsprechen.

Außerdem enthält die Klasse Methoden um Daten in die Komponenten der Oberfläche zu laden oder eingegebene Daten dort wieder heraus zu holen, um diese dann intern weiter verarbeiten zu können. Hierfür werden Getter und Setter für die Einzelnen sich in der Oberfläche befindlichen komponenten implementiert werden, um diese auch von außen ansprechen zu können.

Die Gui Klassen der Challenge Activitys werden einen Setter beinhalten, welcher hervorzuheben ist, da hier nicht ein String übergeben wird, der dann irgendwo als text gesetzt wird, sondern eine komplette klasse, die Challenge Klasse, aus der dann in der Gui Klasse alle informationen gezogen werde, die in der Oberfläche gebraucht werden.

Im Anschluss daran soll die **Data** klasse instanziiert werden, sie wird alle für die Activity wichtigen Daten enthalten, außerdem werden dort die Übergabe Parameter der Letzten Activity angenommen und zwischen gespeichert werden, Dort wird sich in den Meisten fällen auch eine Zugriffsmöglichkeit auf die Globale klasse befinden. In dieser Klasse müssen in manchen fällen auch quasie durchgriffe auf funktionen der Globalen Klasse implementiert werden, da der zugriff darauf nur den Data klassen erlaubt sein soll.

In der Data Klasse für die score activity, welche am ende der bearbeitung einer Kartei angezeigt werden soll, müssen einige berechnungen angestellt werden, da in dieser klasse nur die werte für abgearbeitete und richtig beantwortete Challenges gespeichert sind, es aber noch die werte für Falsch beantwortete Challenges und ein prozentsatz für die Richtig beantworteten benötigt werden.

Als nächstes wird die **AppLogic** Klasse dran sein, in welcher sich , wie der Name schon sagt, die Logik der Entsprechenden Activity befinden wird, hier sollen die Methoden implementiert werden, welche dann später durch gewisse Events ausgelöst werden sollen. Dafür sollen ihr beim Erzeugen die Daten und die Gui klasse als Parameter übergeben werden. Diese Klasse ist allerdings nicht nur dafür gedacht auf events zu reagieren, sondern eignet sie sich auch gut um beim öffnen einer Activity erste daten in die Oberfläche zu laden.

Bei einigen Activitis muss in der AAppLogic classe entschieden werden, welche Activity als nächstes aufgerufen werden soll, beispielsweise nach dem bearbeiten einer challenge muss entschieden werden, welche Feedback Activity aufgerufen wird.

Zu Letzt wird die Klasse **Event\_Listener** erzeugt werden, welche auf die Events der Komponenten mit Hilfe der Gui Klasse Reagieren wird und die Entsprechenden Methoden in der AppLogic aufrufen soll, die für das jeweilige Event gedacht sind.

## Daten Klassen

Es wird eine Klasse für sämtliche Konstanten Angelegt, die die App benötigt wird. Diese Konstanten sollen als Static Variablen angelegt werden, damit die Klasse nicht Instanziiert werden muss um auf die Konstanten von außerhalb zugreifen zu können.

Es soll außerdem noch eine Klasse geben, welche Global Daten halten kann, da eigene Klassen schwer mithilfe des Intends weiter zu reichen sind. Außerdem wäre es in manchen fällen aufwendiger, jedesmal bestimmte Daten als Patrameter an die nächste Activity zu übergeben, als diese einfach global zu halten.

Diese Klasse ist Hauptsächlich dafür gedacht, während der bearbeitung einer Kartei, die dafür immer wieder erforderlichen daten bereitzustellen. Dabei geht es Zunächst Um die Liste Der fälligen challeges der Kartei, für welche der Index erhöht werden und mit hilfe dessen die aktuelle challenge angefordert werden kann.

Außerdem kann von dieser Klasse aus über das ApplicationInterface auf die Persistenten daten zu gegriffen werden. Außerdem wird hier noch die anzahl der bereits richtig beantworteten fragen gespeichert werden.

Es soll außerdem eine Datenklasse für Challenges geben, Die die Frage, den challenge typ und die id Verwaltet, außerdem soll eine Liste vom Datentyp Solution implementiert werden, in dem die möglichen antworten pro challenge gespeichert werden.

In diese klasse sollen Funktionen Implementiert werden, um zu prüfen ob eine bestimmte antworet korrekt ist, um sich alle richtigen antworten als text ausgeben zu lassen, um sich die antworten einzelnd ausgeben zu lassen und um die anzahl der Antwortmöglichkeiten für die Challenge herauszufinden.

Der gerade bereits angesprochene Datentyp Solution soll ebenfalls als klasse angelegt werden. Diese klasse soll lediglich den Antworttext und die Information beinhalten, ob diese antwort korrekt ist.

## Navigations Klasse

Es soll eine Klasse für die Navigation zwischen den Activitys Angelegt werden, in welcher statische Methoden zum Starten der Activitys implementiert werden sollen, die Die Übergabe Parameter per Intend weitergeben. Die Methoden sollen statisch sein, damit die Klasse nicht instanziiert werden muss um auf die Funktionen zugreifen zu können.