

Masterarbeit

„Entwicklung einer mobilen Tutor Anwendung „Teach Me“ für Android Betriebssystem– Konzeption, prototypische Umsetzung und Anwendungsvergleich.“

.

Vorgelegt von: Aleksandr Soloninov

Matrikelnummer:

Email: [inf3032@hs-worns.de](mailto:inf3032@hs-worns.de)

Fachsemester: 4

Abgabedatum: 07.10.2019

Zusammenfassung

Abstract

Danksagung

Abkürzungen

Abbildungen

Zeitplan

**Dauer: 6 Monaten (07.04.2019 – 07.10.2019)**

**Bis 08.04.:** Literaturrecherche

**Bis 15.04.:** Konzeption und Entwicklungssprachen Auswahl

**Bis 31.05.:** Implementierung der App

**Bis 24.06.:** Erster App Test und Vergleich an der UniversitätHeidelberg durchzuführen und Feedback sammeln

**Bis 01.07.:** Ergebnisse vonerstem Test zählen und Rohfassung Hauptteil

**Bis 07.07.:** App Evolution

**Bis 01.08.:** Zweiter App Test und Vergleich an der Universität Mainz durchzuführen und Feedback sammeln

**Bis 07.08.:** Ergebnisse von zweitem Test zählenund App Evolution

**Bis 01.09.:** Dritter App Test und Vergleich an der Hochschule Worms durchzuführen Feedback sammeln

**Bis 15.09.:** Vergleich Auswertung + Hauptteil Vervollständigung

**Bis 01.09.:** Verfassung Einleitung und Schlussverfassung

**Bis 14.09.:** Überarbeitung und Korrektur

**Bis 18.09.:** Layout, Titelblatt und restliche Korrektur

**Bis 19.09.:** Druck

**Bis 07.10.:** Abgabe

Inhaltsverzeichnis

Abkürzungen

Abbildungen

Zeitplan

1. **Einleitung**
   1. Problemstellung
   2. Motivation
   3. Aufbau der Arbeit
   4. Forschungsstand
   5. Forschungskonzept
2. **Grundlagen zur Entwicklung einer Android Applications**

2.1 Integrierte Entwicklungsumgebung

2.1.1 Android Studio

2.1.2 Basic for Android

2.1.3 Visual Studio

2.1.4 AIDE

2.2 Mobile Applikationen

2.2.1 Native App

2.2.2 WebApp

2.2.3 Hybride App

2.3 Android Betriebssystem

2.3.1 Android-Anwendungskomponenten

2.3.2 Pattern

2.4 Entwicklung Sprachen für Android Betriebssystem

2.4.1 Java

2.4.2 Kotlin

2.4.4 Scriptsprachen

2.5 Versionierung

1. **Rolle von Tutorien in Studienprozess**

3.1 Sinn und Zweck von Tutoren in Universitäten

3.2 Arten von Tutorien

3.2.1 Orientierungstutorien

3.2.2 Fachtutorien

3.3 Aufgaben der Tutoren und Tutorinnen

1. **Konzeption**

4.1 Analyse bestehender Softwareprodukten

4.2 Mockup

4.3 Prototype

4.4 UI Design

4.5 Funktionale Anforderungen

4.6 Nicht-funktionale Anforderungen

1. **Implementierung**

5.1 Plattform

5.2 Pattern

5.3 Architecture

5.3.1 Komponenten Zusammenhang

5.3.2 Fragen Struktur

5.4 Engine

5.4.1 TextViewEngine

5.4.2 VideoViewEngine

5.4.3 WebEngine

5.4.4 SpeechEngine

5.5 Activities

5.5.1 Base Activity

5.5.2 Main Activity

5.5.3 Search Activity

5.5.4 Question Activity

5.5.5 Question Select Activity

5.5.6 Question Interview Activity

5.5.7 Score Card Activity

5.5.8 Favorite Activity

5.5.8 Settings Activity

1. **Anwendungsvergleich**

6.1 Auswertungs- und Vergleich Dokumentation

6.2 Forschungsversuch an Ruprecht-Karls-Universität Heidelberg

6.2.1 Forschungsversuch am 3.06.2019

6.2.2 Forschungsversuch am 7.06.2019

6.2.3 Forschungsversuch am 17.06.2019

6.2.4 Forschungsversuch am 20.06.2019

6.2.5 App Evolution

6.3 Forschungsversuch an Hochschule Worms

6.3.1 Forschungsversuch am

6.3.2 Forschungsversuch am

6.3.3 Forschungsversuch am

6.3.4 Forschungsversuch am

6.3.5 App Evolution

6.4 Forschungsversuch an Johannes-Gutenberg-Universität Mainz

6.4.1 Forschungsversuch am

6.4.2 Forschungsversuch am

6.4.3 Forschungsversuch am

6.4.4 Forschungsversuch am

6.4.5 App Evolution

6.5 Gesamtauswertung und Kapitelfazit

1. **Ausblick und Fazit**
2. **Literaturverzeichnis**

Anhang

Problemstellung

Der Mensch will sich immer weiterentwickeln, dazu gehört auch sein Wissensstand.

Vor ca. 30 Jahren konnte man sich nicht vorstellen, dass man sich so viel Wissen aneignen kann, ohne dabei die Haustür zu verlassen. Man musste immer in Bibliotheken, Seminaren und Tutorien gehen.

In der Vorlesung wird nur ein Basisstoff vorgetragen und Studenten können nicht immer den Dozenten Fragen stellen, deswegen gibt es seit ca. 10 Jahren an vielen Universitäten extra „Tutorien-Kurse“ für Studenten. Meist wird dort nichts neues gelernt und die Tutorien können auch nicht immer alle Fragen beantworten oder viele Studenten sind schon berufstätig und haben deshalb gar keine Zeit einen Tutor zu besuchen. Dazu kommt noch in einigen Fällen die Sprachbarriere der Studierenden. Internet und Google Suchmaschine bieten vielartige Möglichkeiten, um brauchbare und verwendbare Information zu finden, nur benötigt man dafür viel Zeit. Mobile Endgeräte entwickeln sich und bieten weitere Möglichkeiten mit einer App Zeit und Sprachbarrieren zu reduzieren.

Motivation

Mit der Entwicklung der Technologie müssen traditionelle Lehrmethoden überarbeitet werden. Dieser Trend verursacht eine gemischte Reaktion. Einerseits gibt es Bedenken, dass neue Technologien Lehrer in der Regel ersetzen werden. Andererseits wird die Technologie nur bestehende Probleme beim Lernen lösen.

In den letzten Jahren hat sich gezeigt, dass moderne Geräte als Unterhaltungsgeräte bezeichnet werden. Ihnen stehen vermeintlich ernstere gewohnheitsmäßige Lehrmethoden gegenüber. Tatsächlich sind digitale Geräte längst zur alltäglichen Realität geworden. Darüber hinaus sind sie für die jüngere Generation bekannter und verständlicher als die Lehrmaterialien, die ältere Menschen gewohnt sind. Durch den Einsatz von Tablets und Smartphones sowie Lernspielen wird der Lernprozess sogar visueller.

Forschungskonzept

Die folgenden Fragen sollen beantwortet werden:

* Inwiefern lässt sich eine mobile Tutor App als ein Tutor dienen?
* Kann eine mobile Tutor App einen realen Tutor oder Tutorin ersetzen?
* Ist Programmiersprache Java immer noch gut für Android Entwicklung anwenden?

Um den Fragen nachzuzeichnen, wird die App „Teach Me“ an die Hochschule Worms, Universität Heidelberg und Universität Mainz getestet. Hierbei handelt es sich um eine Live App zu testen und einen Fragenbogen mit Feedback auszuführen. Die Live Test wird mit Hilfe von folgenden fünf Android Geräten durchgeführt:

* Samsung Galaxy Note 10+
* Samsung Galaxy Note 8
* Samsung Galaxy S7
* Lenovo Tab 10
* Huawei P

Im Rahmen der Masterarbeit werden die folgenden Methodik-Schritte durchgeführt:

1. Eine Analyse von Fragenbogen und Feedback wird durchgeführt, um zu untersuchen, wie sich die App durch das Lernen helfen kann.
2. Wie wurde die App mit Benutzer kommuniziert und wie wurde auf die „Feedback“ von dem Benutzer aufgegriffen?

Zielsetzung und Erkenntnisinteresse

Das Ziel dieser Masterarbeit ist es, eine Konzeption und eine Implementierung einer nativen Android App „Teach Me“ für die Informatik Studierenden an vielen Universitäten, die als ein Tutor dienen soll, zu entwickeln. In der App kann man verschiedene Kurse auswählen und die dazugehörige Information lesen. Mit einem Quiz und Q&A in der App können Studierende ihr Lernwissen überprüfen. Dazu kommen folgenden Aufgaben der Arbeit:

* Eine Analyse das Android Betriebssystem durchzuführen.
* Eine Analyse von bereits bestehenden Entwicklungssprachen für Android Betriebssystemen durchzuführen.
* Eine Konzeption und Implementierung einer nativen Android App „Teach Me“.
* Ein Anwendungsvergleich von „Teach Me“ und einem „Real-Tutor“ durch Informatik Studierende an die Ruprecht-Karls-Universität Heidelberg, Johannes-Gutenberg-Universität Mainz und Hochschule Worms durchzuführen.

Forschungsstand

Durch die allgegenwärtige Verbreitung mobiler Geräte interagieren Menschen unterschiedlich mit Inhalten und der Welt. Durch die steigende Produktivität von Smartphones, Smartwatches und Tablets ermöglicht das mobile Lernen den Studenten den Zugriff auf Lernstoff von überall her, häufig von mehreren Geräten.

Google hat eine Reihe von Strategien für mobile Anwendungen implementiert, die das Wachstum der Branche vorantreiben. Die Reihenfolge, in der Suchergebnisse angezeigt werden, hängt davon ab, ob die Website für mobile Geräte angepasst ist. Google gab außerdem seine Absicht bekannt, seinen Suchindex in den Hauptindex für mobile Geräte und PCs zu unterteilen. [[1]](#page8)

Eine Studie von McGraw-Hill Education und Hannover Research unter mehr als 2.600 Studenten zeigte, dass etwa zwei Drittel der Befragten ihr Smartphone für Bildungszwecke nutzen. [[2]](#page8)

Ein Experiment an der Middlesex University, bei dem mobiles Lernen im Anatomieunterricht bei Erststudierenden eingesetzt wird. Die Studenten der Versuchsgruppe verwendeten eine spezielle Anwendung, die die Struktur der Muskeln und des Skeletts in 3D auf iPad-Geräten anzeigt. Ein Quiz und Spielelemente wurden in die App integriert, um die Aufmerksamkeit zu erhalten. Die Studenten fanden das neue Format „lustig“ und sagten, dass sie es mehr mögen als traditionelle Vorträge. Studenten, die das iPad verwenden, erhielten im Durchschnitt auch höhere Noten. [[3]](#page8)

1. **Grundlagen zur Entwicklung einer Android Applications**

Android предлагает разработчикам массу возможностей: это универсальная, открытая платформа, используемая миллионами пользователей по всему миру, с удивительно простой в использовании площадкой для распространения приложений. Существует множество инструментов для Android-разработчиков, которые помогут быстро начать работу. Но еще лучше то, что с каждым годом инструментов становится все больше, а их эффективность постоянно повышается.

* 1. **Integrierte Entwicklungsumgebung**

Integrierte Entwicklungsumgebung– интегрированная среда разработки программного обеспечения. это программа позволяющих сделать работу программиста более удобной и продуктивной.

**Требования к IDE**:

* Подсветка синтаксиса языка и нумерация строк.
* Функция завершения написания кода и показа параметров.
* Отладка приложения.
* Возможность интеграции с системой контроля версий кода.

IDE — это место, в котором разработчик проводит большую часть своего времени, по этому выбор должен быть сделан правильный.

* + 1. **Android Studio**

[Android Studio](https://developer.android.com/studio/) Это официальная IDE (интегрированная среда разработки) для Android, созданная компанией Google. Именно поэтому Android Studio является номером один для разработчиков, которые хотят создавать приложения, в соответствии с Material дизайном Google и доступом к расширенным функциям платформы.



Abbildung $: Android Studio 3.4.1 mit Geöffneter Projekt „Teach Me“

IDE — это место, в котором разработчик проводит большую часть своего времени: Android Studio выступает в качестве редактора для выбранного вами языка программирования (он поддерживает Java, C++, а также Kotlin, который с недавних пор тоже стал официальным языком Android), компилятора, который может создавать APK файлы и файловой системы для организации вашего проекта. Кроме этого он включает в себя редактор XML и расширенный редактор макетов. Android Studio предлагает целый набор дополнительных инструментов, некоторые из которых мы рассмотрим в данной статье. К счастью, большинство из них теперь можно скачать единым пакетом. По сути, этот пакет идет в комплекте с Android SDK, но вам все равно придется отдельно загрузить и установить Java JDK.

# AVD Manager

Инструмент [AVD Manager](https://developer.android.com/studio/run/managing-avds) идет в комплекте с Android Studio. Аббревиатура AVD расшифровывается как “Android Virtual Device”, поэтому, по сути, это эмулятор для запуска приложений Android на вашем компьютере. Это очень полезный инструмент, который позволяет вам тестировать свои приложения, без необходимости устанавливать их на физические устройства. Что еще более важно, AVD Manager позволяет создавать множество эмуляторов с различными размерами экрана, спецификациями и версиями Android. Вы сможете увидеть, как будет выглядеть ваше творение на любом устройстве, и тем самым обеспечите поддержку среди самых популярных гаджетов. Производительность инструмента постоянно улучшается, особенно с режимом fast virt, который запускает на вашем компьютере версию Android от Intel и устраняет необходимость эмуляции уровня instruction.

Abbildung $: ADV Manager

# Android Device Monitor

Еще один встроенный инструмент — [Android Device Monitor](https://developer.android.com/studio/profile/monitor), который позволяет проводить мониторинг вашего физического или виртуального устройства во время его работы, и, взамен, получать информацию о том, сколько процессов выполняется по потоку, сетевую статистику, LogCat и т.д. Данный инструмент отлично подходит для тестирования производительности ваших приложений.

# Android Debug Bridge

[ADB](https://developer.android.com/studio/command-line/adb) — это инструмент командной строки, с помощью которого можно копировать файлы на устройство и обратно, устанавливать и удалять приложения, выполнять резервное копирование и восстановление на всех устройствах на базе Android, в том числе и на виртуальном.

**2.1.2 Basic for Android**

[B4A](https://www.b4x.com/) (или Basic for Android) — самый малоизвестный инструмент для разработки Android-приложений от компании Anywhere Software, специализирующейся на концепции быстрой разработки приложений (RAD). Как следует из названия, B4A — это IDE и интерпретатор, который позволяет разработчикам создавать приложения с использованием языка программирования BASIC**.** Для тех, кто не знаком с BASIC — это процедурный язык программирования, который читается практически как обычный английский язык. Несмотря на то, что B4A — это проект, разработанный всего одним человеком, ему удалось соединить в себе множество полезных расширенных функций, таких как: беспроводная отладка через Bluetooth, визуальный редактор для добавления и упорядочивания представлений и т.п. Данное ПО не бесплатно.

**@get\_IMAGE\_B4A**

С помощью B4A вы можете делать практически все то же самое, что вы делаете при помощи Java, но гораздо быстрее и с меньшим количеством шаблонов. Любое понижение производительности, по сравнению с другими IDE, минимально. Тем не менее, вам все еще стоит изучить официальный метод создания приложений, особенно если вы хотите использовать конкретные библиотеки (но стоит упомянуть, что библиотеки Java могут быть обернуты для Basic4Android).

* + 1. **Visual Studio**

Visual Studio — это IDE от Microsoft, поддерживающий ряд языков, включая C#, VB.net, JavaScript и многое другое. С помощью фреймворка Xamarin, который входит в Visual Studio, можно создавать кроссплатформенные приложения с помощью C#, а затем тестировать их на нескольких устройствах, подключенных к облаку. Это хороший и бесплатный выбор, если вы планируете выпустить приложение и для Android, и для IOS, но не горите желанием писать свой код дважды. Также он является отличным выбором для тех, кто уже знаком с C# и/или Visual Studio. Минусом является то, что Xamarin неудобен в использовании Java библиотек и, как и с любой другой альтернативой Android Studio, вы теряете поддержку Google и расширенные встроенные функции.

**@get\_IMAGE\_VISUAL\_STUDIO\_2019**

**2.1.4 AIDE**

Наверное вы уже успели заметить, что инструменты для разработки приложений под Android любят аббревиатуры. [AIDE](http://www.android-ide.com/) расшифровывается как “Android IDE” и он уникален тем, что работает на самом Android. Это означает, что вы можете создавать приложения, используя свой телефон или планшет, а затем тестировать их на этом же устройстве. IDE очень хорошо работает с Samsung DeX

**@get\_IMAGE\_AIDE**

AIDE не хватает некоторых функций из Android Studio и у него нет никакого реального преимущества перед более функциональными IDE для разработки под Android. Он пригоден только для изучения Java и Android SDK, так как предоставляет возможность читать учебник по программированию и, в тот же самый момент, проверять оттуда код в режиме реального времени. Это настоящая находка для тех, кто только-только начал разрабатывать собственные мобильные проекты. AIDE не бесплатен и, независимо от того, хорошее о нем общественное мнение или нет — его стоит купить и попробовать самостоятельно.

* 1. **Mobile Applikationen**

Разработчик может выбрать один из трех вариантов мобильных приложений:

* Native App
* WebApp
* Hybride App

Выбор подходящей модели **мобильного приложения** — это очень важный этап в его разработке, на который влияют несколько факторов, таких как техническая оценка разработчиков; потребность в доступе к информации на устройстве; влияние скорости интернета на приложение; одно- или много платформенное ли приложение.

Давайте проанализируем каждый тип, рассмотрим преимущества и недостатки, и определимся, какой из них будет лучшим выбором в той или иной ситуации и лучше всего удовлетворит потребности при создании мобильного приложения.

**2.2.1 Native App**

Под нативным мы подразумеваем **мобильное приложение**, которое создается для определенной платформы и непосредственно устанавливается на устройство пользователя (занимая определенный объем памяти). Такие приложения пользователь загружает через магазин приложений той или иной платформы, такой как Play Store для Google и Apple App Store для iOS.

С нативными приложениями компании могут изготовить приложение согласно индивидуальным запросам, чтобы затем пользователю было удобно им пользоваться, в дополнение к веб-сайту или другому каналу, которым он уже привык пользоваться. Эта целостность и является существенным преимуществом нативных приложений.

Некоторые другие важные преимущества нативных приложений:

* - Обозначение геолокации позволяет компаниям подстраивать свои программы лояльности или промоакции. Потребители могут получать уведомления, когда они находятся возле физических магазинов, или имеют возможность получить региональную скидку.
* - Данные действий (или бездействия) пользователя могут быть легко собраны и проанализированы, таким образом облегчая оценку эффективности всего приложения либо его отдельных функций.
* **- Нативные приложения**, как правило, работают и «чувствуются» лучше. Веб-приложения иногда создаются для имитации нативных, но они ограничиваются скоростью интернета и возможностями дизайна.

И возможные недостатки:

* - Нативные приложения зачастую более дорогие в разработке, особенно для компаний, которым нужны приложения на кроссплатформенных ОС
* - Нативные приложения должны быть одобрены каждым магазином приложений, а процесс привлечения внимания к нему пользователей может быть сложным (если это не приложение для внутреннего пользования в компании)

**2.2.2 WebApp**

Как вы уже могли догадаться, эти приложения работают через веб-браузер на устройстве пользователя. Эти приложения по сути являются индивидуализированными вебсайтами, которые сделаны таким образом, чтобы выглядеть и использоваться как нативные приложения, но на самом деле они не находятся на устройстве пользователя. Их можно сравнить с облачным хранилищем в сравнении с данными, которые хранятся на жестком диске компьютера. При хорошей, качественной разработке, которая включает в себя подбор размеров и прокрутку, веб-приложения часто работают подобно нативным приложениям.

Вот некоторые ключевые преимущества веб-приложений:

* - Приложения на веб-основе легче поддерживаются и они могут функционировать на платформе с любой ОС
* - Разработчики могут предлагать приложения без необходимости их утверждения какими-либо магазинами приложений
* - Более быстрая разработка циклов с использованием CSS, HTML и JavaScript

И несколько минусов:

* - У веб-приложений нет доступа к устройству пользователя. Несмотря на то, что иногда было бы удобно, это ограничивает многие функции, которые используются в нативных приложениях для более персонализированного использования
* - Пользователи должны использовать их через сеть, что значительно снижает контроль безопасности
* - Поиск приложения может быть затруднительным, так как не существует магазина приложений с каталогом и функцией поиска в нем
  + 1. **Hybride App**

Гибридные приложения являются чем-то средним между нативными и веб-приложениями. Фактически они создаются так, чтобы выглядеть и использоваться как нативные приложения. Их также устанавливают на телефон пользователя и их можно найти в магазинах приложений. Различие заключается в том, что они обязательно должны размещаться в рамках нативного приложения и созданы, чтобы работать через WebView, и таким образом они могут получать доступ к информации на устройстве пользователя для больших возможностей.

Дополнительные преимущества гибридных приложений:

* - Гибридные приложения обладают наибольшей функциональностью и персонализацией для пользователя
* - Разработчики не ограничены одной платформой, вместо этого они могут создать гибридное приложение, которое будет работать с несколькими платформами (в случае работы как нативное приложение)
* - Гибриды — хорошая опция для разработчиков, которые создают визуально насыщенные приложения, например, игры (которые не будут хорошо работать в виде веб-приложений)

В любом случае, есть некоторые недостатки, о которых стоит подумать при выборе гибридного приложения:

* - Слишком сложные приложения лучше всего делать нативными
* - Разработка требует дополнительных времени и усилий (по сравнению с веб-приложениями), чтобы такое приложение выглядело и ощущалось пользователем как нативное
* - Магазины приложений могут отклонять гибридные приложения, которые работают недостаточно плавно
  1. **Android Betriebssystem**

**Android** — операционная система для смартфонов, интернет-планшетов, электронных книг, цифровых проигрывателей, наручных часов, игровых приставок, нетбуков, смартбуков, очков Google, телевизоров и других устройств. Основана на ядре Linux и собственной реализации виртуальной машины Java от Google. Изначально разрабатывалась компанией Android, Inc., которую затем купила Google. Впоследствии Google инициировала создание альянса Open Handset Alliance (OHA), который сейчас занимается поддержкой и дальнейшим развитием платформы. Android позволяет создавать Java-приложения, управляющие устройством через разработанные Google библиотеки. Android Native Development Kit позволяет портировать библиотеки и компоненты приложений, написанные на [С](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F))и и других языках.

На начало 2019 года, по исследованиям с разных источников удалось выяснить, что в среднем доля смартфонов с ОС Android на рынке смартфонов превысила 86% от общего количества проданных смартфонов в мире. Всего за 2016 год было продано более 344 миллионов устройств.

**2.3.1 Anwendungskomponenten**  
Android-приложения состоят из следующих частей:

* **Activity**/Деятельность (далее Активити) — представляет собой схему представления Android-приложений. Например, экран, который видит пользователь. Android-приложение может иметь несколько активити и может переключаться между ними во время выполнения приложения.
* **Views**/Виды — Пользовательский интерфейс активити, создаваемый виджетами классов, наследуемых от «android. view.View». Схема views управляется через «android.view.ViewGroups».
* **Services**/Службы — выполняет фоновые задачи без предоставления пользовательского интерфейса. Они могут уведомлять пользователя через систему уведомлений Android.
* **Content Provider**/Контент-провайдеры — предоставляет данные приложениям, с помощью контент-провайдера Ваше приложение может обмениваться данными с другими приложениями. Android содержит базу данных SQLite, которая может быть контент-провайдером
* **Intents**/Намерения (далее Интенты) — асинхронные сообщения, которые позволяют приложению запросить функции из других служб или активити. Приложение может делать прямые интенты службе или активити (явное намерение) или запросить у Android зарегистрированные службы и приложения для интента (неявное намерение). Для примера, приложение может запросить через интент контакт из приложения контактов (телефонной/записной книги) аппарата. Приложение регистрирует само себя в интентах через IntentFilter. Интенты — мощный концепт, позволяющий создавать слабосвязанные приложения.
* **Broadcast Receiver** — принимает системные сообщения и неявные интенты, может использоваться для реагирования на изменение состояния системы. Приложение может регистрироваться как приемник определенных событий и может быть запущено, если такое событие произойдет.

Жизненный цикл приложения в Android жёстко контролируется системой и зависит от нужд пользователя, доступных ресурсов и т. д. Например, пользователь хочет запустить браузер. Решение о запуске приложения принимает система. Хотя последнее слово и остаётся за системой, она подчиняется определённым заданным и логическим правилам, позволяющим определить, можно ли загрузить, приостановить приложение или прекратить его работу. Если в данный момент пользователь работает с определённым окном, система даёт приоритет соответствующему приложению. И наоборот, если окно невидимо и система решает, что работу приложения необходимо остановить, чтобы мобилизовать дополнительные ресурсы, будет прекращена работа приложения, имеющего более низкий приоритет. В Android ресурсы более ограниченны, поэтому Android более жёстко контролирует работу приложений.

Основные методы жизненного цикла приложения

* protected void onCreate()
* protected void onStart()
* protected void onRestart()
* protected void onResume()
* protected void onPause()
* protected void onStop()
* protected void onDestroy();



**2.3.2 Pattern**

При разработке сложных приложений можно столкнуться с проблемами, которые, вероятно, возникали раньше и уже имеют большое количество решений. Такие решения называются **паттернами**. Как правило, говорят о паттернах дизайна и паттернах архитектуры. Они упрощают разработку приложений, поэтому целесообразно их использовать, если такая возможность есть.

Приложение, Teach Me, было разработано используя паттерн архитектуры MVP.

**Model View Presenter**

**@get\_IMAGE\_MVP**

MVP паттерн разработки для ОС Android, предлагающий разбивать приложение на следующие три части:

1. Model — является оберткой для полученных данных. При этом особой разницы откуда данные быть не должно — данные сетевых запросов или данные взаимодействия пользователя с пользовательским интерфейсом. Хорошее место для внедрения кэшей. Хорошей практикой является для каждого ответа сервера создавать уникальную модель, для уменьшения пересечений и последующих проблем при изменений api.[8-10]
2. Presenter — устанавливает связь между обработкой данных, получаемых из Model и вызовом методов у View, реализуя тем самым реакцию компонентов пользовательского интерфейса на данные. Методы Presenter вызываются из методов жизненного цикла activity/fragment и часто симметричны им .[8-10]
3. View - отображает полученные данные из Presenter. В правильной реализации объект View не имеет представления о полученных данных из вне, он должен только отображать что требует от него Presenter. View может быть любая Activity или Fragment в приложении под ОС Android.[8-10]

Model View Presenter должны представлять из себя интерфейсы для большей гибкости модификации кода.

Данная архитектура имеет ряд плюсов по сравнению со стандартной схемой написания приложения одним из которым является хорошее покрытие тестами, что важно для реализации приложения, а также гибкости. Приложение не знает что происходит до того как отразить элемент на экране, вся логика закрыта от View. Два пункта учтены по созданию хорошей архитектуре. Так как MVP разделяют логику и отображение элементов, а также если приложение имеет клиент серверную структуру разделяет и запросы к серверу. С помощью MVP можно также достичь и довольно простого и понятного кода, это третий пункт хорошей архитектуры приложения. Кроме этого MVP делает можно сказать свои собственные блоки для каждого окна, который видит пользователь, это означает что система, приложение, спокойно расширяется без всяких конфликтов.[12]

**2.4 Entwicklung Sprachen für Android Betriebssystem**

**2.4.1 Java**

Официальный язык программирования, поддерживаемый средой разработки Android Studio. По данным [ежегодного опроса ресурса Stackoverflow](https://insights.stackoverflow.com/survey/2018/#technology), в 2018 году Java вошёл в пятёрку самых популярных языков программирования.

На Java ссылается большинство официальной документации Google, а найти платные и бесплатные библиотеки и руководства не составит труда — их великое множество.

К сожалению, сложность Java мешает программировать на нём всем желающим. Как у объектно-ориентированного языка программирования у него куча особенностей в виде конструкторов классов, исключений, приводящих к падению приложений во время работы и других моментов, которые всегда необходимо учитывать при разработке. Впрочем, код на Java легко читается и структурируется, особенно при соблюдении принятых стандартов его оформления.

При разработке на Java под Android используются не только Java-классы, содержащие код, но также файлы манифеста на языке XML, предоставляющие системе основную информацию о программе, и системы автоматической сборки Gradle, Maven или Ant, команды в которых пишутся на языках Groovy, POM и XML соответственно; по умолчанию в проектах используется Gradle, а на начальных этапах обучения разработке на Java править файлы, написанные на Groovy, практически не придётся. Для вёрстки UI-части обычно также используется язык XML.

Android Studio, в декабре 2014 года признанная Google официальной средой разработки под ОС Android, совершенствуется год от года, чем облегчает жизнь Android-разработчикам. Такие её возможности, как визуальный UI-редактор и автодополнение кода, помогают сделать процесс разработки более комфортным.

**2.4.2 Kotlin**

Язык был официально представлен в мае 2017 года на Google I/O и позиционируется Google как второй официальный язык программирования под Android после Java, только чуть более простой для понимания. Знания Java необходимы здесь, чтобы понимать принципы работы Kotlin, общую структуру языка и его особенности. Многие разработчики считают Kotlin обёрткой над Java и рекомендуют изучать его только после того, как вы почувствуете уверенность в своих знаниях Java.

Kotlin совместим с Java и не вызывает снижения производительности и увеличения размера файлов. Отличие от Java в том, что он требует меньше служебного, так называемого boilerplate-кода, поэтому более обтекаемый и лёгкий для чтения. Его создателям удалось избежать nullpointexceptions, а компиляция больше не прерывается из-за мелочей вроде забытого знака «;».

**2.4.4 Scriptsprachen**

**Lua**

Lua – старый скриптовый язык, который изначально создавался в качестве дополнения для программ, написанных на более сложных языках: C, VB.NET и т.д. В этом языке есть некоторые особенности, которые выделяют Lua из ряда подобных ему – к примеру, начало массивов с 1 вместо 0, или отсутствие нативных классов.

Таким образом, для определённых задач Lua можно использовать в качестве основного языка программирования. Лучший тому пример – SDK Corona. При помощи Corona можно создавать мощные, богатые по функциональности приложения с возможностью развёртывания на Windows, Mac, Android, iOS, и даже Apple TV + Android TV. В Corona также встроены возможности для монетизации, плюс – это приличный по объёмам рынок, где можно отыскать полезные в работе плагины.

## **HTML5 + CSS + JavaScript**

Эта тройка языков, когда-то созданных для разработки фронт-энд приложений в веб-среде, с тех пор переросла в нечто большее. Теперь инструментов HTML5, CSS и JavaScript вполне достаточно для создания самых разных приложений и для мобильных устройств, и для классических ПК. По сути, программист создаёт веб-приложение, способное использовать всю мощь и магию оффлайн-платформ.

Для создания Android-приложений этим способом можно использовать возможности Adobe Cordova – это фреймворк с открытым исходным кодом, который также поддерживает операционные системы iOS, Windows 10 Mobile, Blackberry, Firefox, и многие другие. Однако, какой бы полезной ни была Cordova, для создания в ней сколь-нибудь приличного приложения требуется серьёзная работа. Поэтому многие программисты отдают предпочтение проекту Ionic Framework (который использует «Кордову» для развертывания на различных платформах).

Есть и другая возможность: использование библиотеки React Native. Её можно развернуть на Android, iOS. Эту библиотеку используют специалисты Facebook, Instagram и других крупных компаний, поэтому можно положиться на её надёжность.

**2.5 Git Versionierung**

Git – это разновидность Version Control System. А VCS – это программа для работы с постоянно изменяющейся информацией. Такое ПО может хранить множество версий одного и того же файла и возвращаться к более раннему состоянию.

Дипломная работа использует сервис хостинга Github.com и проект Teach Me доступен по адрессу: https://github.com/AuroraSyN/Teach-Me

Github.com это веб-сервис проектов с использованием системы контроля версий git, а также как социальная сеть для разработчиков. Пользователи могут создавать неограниченное число репозиториев, для каждого из которых предоставляется wiki, система issue tracking-а, есть возможность проводить code review и многое другое.

**3. Rolle von Tutorien in Studienprozess**

**3.1 Sinn und Zweck von Tutoren in Universitäten**

Ein Tutorium ist eine unterstützende Veranstaltung. Hier können Studenten gemeinsam mit einem Tutor den Stoff einer Lehrveranstaltung wiederholen und vertiefen. Ein Tutorium ist in den meisten Fällen freiwillig, kann aber von einem Dozenten vorgeschrieben werden. Wird eine solche Veranstaltung angeboten, sollte man die Chance ergreifen und diese besuchen. Oft dienen Tutorien auch dazu, gemeinsam Hausaufgaben zu erarbeiten, während der Tutor den Anwesenden hilft und ihnen mit Rat und Tat zur Seite steht.

Ein Tutor ist meist selbst Student in einem höheren Fachsemester. Er unterstützt Dozenten als studentische Hilfskraft und erleichtert anderen Studenten mit seinem Wissen den Start ins Studium.

**3.2 Arten von Tutorien**

Tutoren können in einer Vielzahl von Tutorien tätig sein, die jeweils unterschiedliche Ziele verfolgen. Dabei kann es sich um persönliche Betreuung oder Gruppenbetreuung handeln. So ﬁndet sich in Großbritannien z.B. das ,Personal Tutoring System‘, bei dem jedem Studierenden ein Hochschullehrer zugewiesen wird, der ihm als Ansprechperson dienen soll. In Anbetracht der hohen Studierendenzahlen wird im universitären Umfeld in Deutschland eher auf die Betreuung von Gruppen gesetzt. Dabei sind vor allem studentische Tutoren im Einsatz. Im Fachbereich 05 der Universität Kassel sind vor allem zwei Arten von Tutorien zu ﬁnden: Orientierungstutorien und Fachtutorien.

**3.2.1 Orientierungstutorien**

Orientierungstutorien erstrecken sich meist über drei Tage vor Beginn des ersten Semestersundsollendazudienen,”Studienanf¨angerneine(imweitestenSinne)sozialeOrientierung an der Hochschule, im Studienfach und am Hochschulort“ zu geben, um somit den Studienanfang zu erleichtern (Knauf, 2005, 1). Diese können sich an die Studienanfänger generell, oder aber auch nur an bestimmte Zielgruppen (z.B. ausländische Studenten) richten (Stemmler, 1997). Hierbei steht unter anderem der Gedanke im Mittelpunkt, die Hochschule nicht nur als Lernort sondern auch als Lebensraum zu verstehen. Der Beginn des Studiums bringt viele neue Anforderungen mit sich. Neben den veränderten fachlichen Anforderungen stehen die Erwartungen an selbstverantwortliche Lernkompetenzen und zielorientiertes Arbeiten. Darüber hinaus ist für viele mit dem Eintritt in eine Hochschule zudem ein Wohnortswechsel verbunden. Einerseits muss der Studierende nun lernen, für sich und seine akademischen Leistungen selbst Verantwortung zu tragen, indem er sich z.B. selbst seinen Stundenplan organisiert und für die Erstellung und Einreichung der Leistungsnachweise verantwortlich ist. Andererseits muss er auch lernen, außerhalb der Universität zu Recht zu kommen und sich an das selbstständige Leben in einer eigenen Wohnung in einer fremden Stadt gewöhnen. Der Fachbereich hat bis jetzt gute Erfahrungen mit Orientierungstutorien gemacht. Dort wurde Studienanfängern in größeren Gruppen neben der fachlichen Einweisung auch Informationen über die Stadt und das Leben als Student gegeben, um ihnen so die Umstellung zu erleichtern.

**3.2.2 Fachtutorien**

Fachtutorien werden in der Regel als Begleitung zu Kernveranstaltungen im ersten Studienjahr angeboten und dienen dazu, das fachliche Lernen im Rahmen einer kleineren Gruppe zu unterstützen. Bei Fachtutorien kann es sich um separate Tutorien handeln oder um integrierte Veranstaltungen, bei denen sich Lehrender, Tutor und Studierende zur selben Zeit im selben Raum beﬁnden. Fachtutorien ermöglichen den Studenten zudem erforderliche Grundfähigkeiten zu erwerben, aufzubauen und zu trainieren: ”Viele Lernangebote gehen von einer ﬁngierten Lernselbstständigkeit der Studierende aus, ohne sie entwickeln zu helfen oder Lernziele auszuformulieren“. Was genau in den Tutorien passieren soll, wird vom jeweiligen Dozenten bestimmt. Elementar ist dabei die Zusammenarbeit von Dozent und Tutor, um das Tutorium möglichst gut an die Veranstaltung anzupassen und dem Studierenden den größtmöglichen Lernerfolg zu garantieren. Durch Vorgaben des Dozenten können die Studierenden sich sicher sein, dass die Inhalte des Tutoriums mit den Anforderungen des Dozenten übereinstimmen. Durch Absprachen wird für die Teilnehmer zudem eher der Bezug zur Veranstaltung und somit auch der Sinn des Tutoriums sichtbar. Rückmeldungen der Tutoren können dann hilfreich sein, um dem Dozenten ein Bild über den Wissens- und Kompetenzstand seiner Studierenden zu verschaﬀen, und eventuell besonders problematische Themen noch einmal aufzugreifen.

**3.3 Aufgaben der Tutoren und Tutorinnen**

Neben der eigenen Lehrtätigkeit in einem Tutorium helfen Tutoren unter anderem bei der Vorbereitung von Veranstaltungen, unterstützen Professoren und betreuen ausländische Studenten.

**#ToDO**

**4. Konzeption**

**4.1 Analyse bestehender Softwareprodukten**

**4.2 Mockup**

**4.3 Prototype**

**4.4 UI Design**

**4.5 Funktionale Anforderungen**

**4.6 Nicht-funktionale Anforderungen**

**5. Implementierung**

**5.1 Plattform**

**5.2 Pattern**

**5.3 Architecture**

**5.3.1 Komponenten Zusammenhang**

**5.3.2 Fragen Struktur**

**5.4 Engine**

**5.4.1 TextViewEngine**

**5.4.2 VideoViewEngine**

**5.4.3 WebEngine**

**5.4.4 SpeechEngine**

**5.5 Activities**

**5.5.1 Base Activity**

**5.5.2 Main Activity**

**5.5.3 Search Activity**

**5.5.4 Question Activity**

**5.5.5 Question Select Activity**

**5.5.6 Question Interview Activity**

**5.5.7 Score Card Activity**

**5.5.8 Favorite Activity**

**5.5.8 Settings Activity**

**6. Anwendungsvergleich**

**6.1 Auswertungs- und Vergleich Dokumentation**

**6.2 Forschungsversuch an Ruprecht-Karls-Universität Heidelberg**

**6.2.1 Forschungsversuch am 3.06.2019**

**6.2.2 Forschungsversuch am 7.06.2019**

**6.2.3 Forschungsversuch am 17.06.2019**

**6.2.4 Forschungsversuch am 20.06.2019**

**6.2.5 App Evolution**

**6.3 Forschungsversuch an Hochschule Worms**

**6.3.1 Forschungsversuch am**

**6.3.2 Forschungsversuch am**

**6.3.3 Forschungsversuch am**

**6.3.4 Forschungsversuch am**

**6.3.5 App Evolution**

**6.4 Forschungsversuch an Johannes-Gutenberg-Universität Mainz**

**6.4.1 Forschungsversuch am**

**6.4.2 Forschungsversuch am**

**6.4.3 Forschungsversuch am**

**6.4.4 Forschungsversuch am**

**6.4.5 App Evolution**

**6.5 Gesamtauswertung und Kapitelfazit**

**7. Ausblick und Fazit**

**8. Literaturverzeichnis**