**Основы Android 03.1:Отладчик**

Последнее обновление: 19 февраля 2021 г.

Вступление

В предыдущих практических [Log](https://developer.android.com/reference/android/util/Log.html" \t "_blank)занятиях вы использовали класс для печати информации в системном журнале, который появляется на панели **Logcat** в Android Studio при **запуске** вашего приложения. Добавление операторов журнала в ваше приложение - один из способов найти ошибки и улучшить работу вашего приложения. Другой способ - использовать отладчик, встроенный в Android Studio.

В этом практическом занятии вы узнаете, как отлаживать приложение в эмуляторе и на устройстве, устанавливать и просматривать точки останова, выполнять пошаговое выполнение кода и изучать переменные.

Что вы уже должны знать

Ты должен быть способен:

* Создайте проект Android Studio.
* Используйте редактор макета для работы с EditTextи Buttonэлементами.
* Создайте и запустите свое приложение в Android Studio как на эмуляторе, так и на устройстве.
* Прочтите и проанализируйте трассировку стека, включая последнее включение, первое отключение.
* Добавьте операторы журнала и просмотрите системный журнал ( панель **Logcat** ) в Android Studio.

Что ты узнаешь

* Как запустить приложение в режиме отладки в эмуляторе или на устройстве.
* Как выполнить пошаговое выполнение вашего приложения.
* Как устанавливать и организовывать точки останова.
* Как проверять и изменять переменные в отладчике.

Что ты будешь делать

* Создайте приложение SimpleCalc.
* Установите и просмотрите точки останова в коде SimpleCalc.
* Пошагово выполняйте свой код по мере его выполнения.
* Изучите переменные и оцените выражения.
* Выявите и устраните проблемы в образце приложения.

<https://github.com/google-developer-training/android-fundamentals-starter-apps-v2/tree/master/SimpleCalc>

SimpleCalc - начальный код

=========================

SimpleCalc - это простое приложение-калькулятор, в котором есть несколько проблем.   
Этот app используется для изучения отладчика Android Studio.

Это также основа для изучения модульного тестирования в SimpleCalcTest.

Предварительные условия

--------------

Для этого приложения вы должны быть знакомы с:

\* Создание проекта Android Studio и работа с представлениями EditText и Button.

\* Создание и запуск вашего приложения в Android Studio как в эмуляторе, так и на устройстве.

\* Добавление операторов журнала и просмотр системного журнала (logcat) в Android Monitor.

25.03.2021

Кодовая база примера на Java, AS 3.4, старые библиотеки и не android **и активити** не AppCompat:

В AS 4.1.3 простраивался с предупреждениями и стартовал и работал

1. Поднял Gradle на 4.1.3 – работает.

2. Преобразовал в AndroidX и в AppCompatActivity – работает

3. Библиотеки подняли до актуальных – работает.

4. Не выбросил не нужное из Gradle

5. Можно преобразовать в Kotlin, должно работать.

6. Можно переписать в JetPack (Buinding, ViewModel и т.д.)

7. Можно JetPack на Kotlin уже сделать с возможностями Kotlin, в т.ч. с функциональным программированием.

8. Можно переписать в мультиплатформу – не умею

9. Можно в модульную структуру – не умею

10. В атомарную программу – не умею

11. Включить ADS, Firebase

12. Можно подготовить его к GP

13. Можно выставить в GP

Предполагаю, что есть потребность у людей с 1-4

Предполагаю, что есть с 5-7

Есть спрос на 8-10

Есть 11-12

**Задача 6:** в конце задачи 1 вы попытались запустить приложение SimpleCalc без значения в одном из EditText элементов, что привело к ошибке. Используйте отладчик для пошагового выполнения кода и точного определения причины возникновения этой ошибки. Исправьте ошибку, которая вызывает эту ошибку.

Часть1:

1. Запустить приложение без отладчика, открыть панель Logcat

2. Одно поле оставить пустым

3. add

4. По трассировке нашла где ломается, откуда зовется

5. Поставила точку останова 137

6. Определила, что причина ошибки getOperandText на второй операнд возвращает пустую строку в 186 строку возвращает пустую строку и загоняет в переменную операнд текст.

7. В 187 строке пустую строку пытаются преобразовать в плавующее число. Что не определено. Поэтому возникает исключение сбой программы, но наверху стоит перехват, поэтому не ломается, а просто выводит Error.

Часть2:

Правильное решение это поставить этот трай кетч 135-143 строки. Но так как нам делать нечего, то вместо пустой строки подсунем 0. И тогда трай кетч возникать не будет. – Реализовано

Решение 2 – решили пустую строку менять на ноль

// Мое первое исправление: Если строка пустая, то будет ноль  
if (operandText.equals("")) {  
 operandText="0";  
}

Решение 3 – решили менять на ноль другим способом

Можно в xml значение поставить по умолчанию ноль, но если его пользователь стер, то все равно будет ошибка.

Решение 4 Можно ноль присваивать раньше, чем возникает ошибка.

Решение 5 Можно покопаться все-таки с EditText, найдется как не допускать пустую строку ввода в Uх

Решение 6 Можно поставить слушатель над этими полями и когда я дам Enter, то туда подставлять ноль прямо в поле.

Решение 7 Подключить Биндинг и тогда я могу обработать поле прямо в XML, т.е. при уходе с поля прямо здесь его проверить, если пусто то поставить ноль

Решение 7 правильное – перейти на материал дизайн и там есть EditText там есть допустимые значения и тогда он будет общаться с пользователем и проверять, что пользователь вообще ввел цифровое значение. Материал дизайн дает возможность не выпускать пользователя из этого поля.

Лучше всего применить 7, но ожидают первое решение

Объявить это Эдит поле обязательным и с пустым не выпустит.

 Резюме

* Просмотрите информацию журнала в Android Studio, щелкнув вкладку **Logcat** .
* Запустите приложение в режиме отладки, щелкнув значок «Отладка» или выбрав « **Выполнить»> «Отладка приложения»** .
* Щелкните вкладку « **Отладка** », чтобы отобразить панель **отладки** . Щелкните вкладку « **Отладчик** » на панели « **Отладка»,** чтобы отобразить панель « **Отладчик»** (если она еще не выбрана).
* На панели **отладчика** отображаются (складываются) **кадры** , **переменные** в определенном кадре и **наблюдения** (активное отслеживание переменной во время выполнения программы).
* Точка останова - это место в вашем коде, где вы хотите приостановить нормальное выполнение вашего приложения для выполнения других действий. Установите или очистите точку останова отладки, щелкнув левую кромку окна редактора сразу же рядом с целевой строкой.

<https://google-developer-training.github.io/android-developer-fundamentals-course-concepts-v2/unit-1-get-started/lesson-3-testing,-debugging,-and-using-support-libraries/3-1-c-the-android-studio-debugger/3-1-c-the-android-studio-debugger.html>

## 9. Узнать больше

Документация по Android Studio:

* [Руководство пользователя Android Studio](https://developer.android.com/studio/intro/index.html)
* [Отладьте свое приложение](https://developer.android.com/studio/debug/index.html)
* [Запись и просмотр журналов](https://developer.android.com/studio/debug/am-logcat.html)
* [Анализировать трассировку стека](https://developer.android.com/studio/debug/stacktraces.html)
* [Android Debug Bridge](https://developer.android.com/studio/command-line/adb.html)
* [Профайлер Android](https://developer.android.com/studio/profile/android-profiler.html)
* [Профайлер сети](https://developer.android.com/studio/profile/network-profiler.html)
* [Профайлер ЦП](https://developer.android.com/studio/profile/cpu-profiler.html)
* [Traceview](https://developer.android.com/studio/profile/traceview.html)

Другой:

* Видео: [отладка и тестирование в Android Studio](https://www.youtube.com/watch?v=2I6fuD20qlY)

## 10. Домашнее задание

В этом разделе перечислены возможные домашние задания для студентов, которые работают над этой лабораторией кода в рамках курса, проводимого инструктором. Инструктор должен сделать следующее:

* При необходимости назначьте домашнее задание.
* Сообщите учащимся, как сдавать домашние задания.
* Оценивайте домашние задания.

Преподаватели могут использовать эти предложения столько, сколько захотят, и могут свободно назначать любое другое домашнее задание, которое они считают подходящим.

Если вы работаете с этой лабораторией самостоятельно, не стесняйтесь использовать эти домашние задания, чтобы проверить свои знания.

## ****Создайте и запустите приложение****

Откройте приложение [SimpleCalc](https://github.com/google-developer-training/android-fundamentals-starter-apps-v2/tree/master/SimpleCalc" \t "_blank) .

1. В MainActivity поместите точку останова в первую строку onAdd()метода.
2. Запустите приложение в отладчике. Выполните операцию добавления в приложении. Выполнение останавливается в точке останова.
3. Щелкните « **Шаг с заходом»,** чтобы шаг за шагом проследить за выполнением приложения. Обратите внимание, что **Step Into** открывает и выполняет файлы из платформы Android, позволяя вам увидеть, как сам Android работает с вашим кодом.
4. Изучите, как изменяется панель **отладки** при пошаговом выполнении кода для текущего кадра стека и локальных переменных.
5. Изучите, как аннотируется сам код на панели редактора при выполнении каждой строки.
6. Щелкните « **Шаг с выходом»,** чтобы вернуться в приложение, если стек выполнения становится слишком глубоким для понимания.

## ****Ответьте на эти вопросы****

### **Вопрос 1**

Запустите приложение SimpleCalc без отладчика. Оставьте один или оба EditTextэлемента пустыми и попробуйте выполнить любой расчет. Почему возникла ошибка?

* java.lang.NumberFormatException: empty String
* W/OpenGLRenderer: Failed to choose config with EGL\_SWAP\_BEHAVIOR\_PRESERVED
* Приложение может слишком много работать над своим основным потоком.
* Емкость кэша кода увеличена до 128 КБ.

### **вопрос 2**

Какую функцию вы выполняете на панели отладки, чтобы выполнить текущую строку, где находится точка останова, а затем остановиться на следующей строке кода? Выбери один:

* **Шаг в**
* **Переступить**
* **Выйти**
* **Продолжить**

### **Вопрос 3**

Какую функцию вы выполняете на панели отладки, чтобы перейти к выполнению вызова метода из текущей строки, где находится точка останова? Выбери один:

* **Шаг в**
* **Переступить**
* **Выйти**
* **Продолжить**

## ****Отправьте свое приложение на оценку****

### **Руководство для грейдеров**

Нет приложения для выполнения этого домашнего задания.