МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ

КАЛИНИНГРАДСКОЙ ОБЛАСТИ

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ

КАЛИНИНГРАДСКОЙ ОБЛАСТИ

ПРОФЕССИОНАЛЬНАЯ ОБРАЗОВАТЕЛЬНАЯ ОРГАНИЗАЦИЯ

«КОЛЛЕДЖ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СТРОИТЕЛЬСТВА»

(ГБУ КО ПОО «КИТиС»)

|  |  |  |
| --- | --- | --- |
| Курсовой проект  допущен к защите  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (дата)  Зам.директора по УМР  \_\_\_\_\_\_\_\_\_\_\_\_\_Павленко Г.Я.  (подпись) |  | Курсовой проект  защищен с оценкой  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оценка)  Руководитель проект  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) |

Пояснительная записка к курсовому проекту

по дисциплине: МДК 01.01 Разработка программных модулей

Тема: Разработка Мобильного приложения «Дробный калькулятор»

Специальность 09.02.07 «Информационные системы и программирование»

|  |  |
| --- | --- |
| Выполнил: | студент 3 курса,  группы ИСп19-2к  Попков Савелий Михайлович  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |
| Руководитель: | Большакова-Стрекалова Анна Викторовна, преподаватель ГБУ КО ПОО КИТиС  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |

Калининград

2022

Содержание

Введение 2

1 Описание предметной области 3

1.1 Аналоги разрабатываемого приложения 4

1.2 Техническое задание 6

1.3 Описание структуры приложения 7

2 Описание разработки приложения 8

2.1 Обоснование средств разработки 12

2.2 Разработка интерфейса 12

2.3 Разработка логики работы приложения. Схема взаимодействия компонентов проекта 14

2.4 Описание переменных, компонентов, классов и подпрограмм 14

3 Тестирование и установка приложения 22

Заключение 22

Список использованной литературы 23

Приложение А. Листинг рабочего кода 24

Приложение Б. Руководство пользователю 31

Введение

С развитием технологий человек стремился создать аппаратуру, позволяющую укомплектовать инструментарий, так необходимый для повседневной жизни человека. В наше время лучше чем смартфон никто не сможет удовлетворить данную потребность человечества. И об одном из таком необходимом инструментарии мы сегодня поговорим. Представьте, что вам нужно посчитать общую сумму покупок, или же просто выполнить домашнее задание по математике, а может быть быстро решить сложное математическое уравнение, в котором у вас нет времени считать обычные арифметические вычисления? Для этого человечество и придумало калькулятор. Быстрое и удобное электронно-вычислительное устройство, прямо у вас в кармане, необходимое на разные случаи жизни.

В данном случае мы будем разрабатывать калькулятор для платформы Android, код будет писаться на языке Java

Java обеспечивает быстроту работы, легкость компиляции, совместимость, интеграционную способность и т.д.

Целью исследования является процесс разработки приложения.

Для достижения поставленной цели необходимо решить ряд задач:

* рассмотреть роль современных приложений в процессе коммуникации с целевой аудиторией;
* исследовать основные организационные, экономические и маркетинговые особенности разработки приложения;
* дать обоснование созданию приложения и изучить этапы его создания

1. Описание предметной области

Java является основой практически для всех типов сетевых приложений и всеобщим стандартом для разработки и распространения встроенных и мобильных приложений, игр, веб-контента и корпоративного программного обеспечения. В мире насчитывается более 9 миллионов специалистов, разрабатывающих приложения на Java, которая позволяет эффективно разрабатывать, внедрять и использовать превосходные приложения и услуги.

Технология Java протестирована, усовершенствована, расширена и проверена участниками сообщества разработчиков Java, архитекторов и энтузиастов. Java позволяет разрабатывать высокопроизводительные портативные приложения практически на всех компьютерных платформах. Доступность приложений в разнородных средах позволяет компаниям предоставлять более широкий спектр услуг, способствует повышению производительности, уровня взаимодействия и совместной работы конечных пользователей и существенному снижению стоимости совместного владения корпоративными и потребительскими приложениями. Java стала незаменимым инструментом для разработчиков и открыла для них следующие возможности:

* написание программного обеспечения на одной платформе и его запуск практически на любой другой платформе;
* создание программ, работающих в веб-браузере и имеющих доступ к веб-службам;
* разработка приложений на стороне сервера для форумов в Интернете, магазинов, опросов, обработки форм HTML и много другого;
* объединение приложений или служб с использованием языка Java для создания высокоспециализированных приложений или служб;
* создание многофункциональных и эффективных приложений для мобильных телефонов, удаленных процессоров, микроконтроллеров, беспроводных модулей, датчиков, шлюзов, потребительских продуктов и практически любых других категорий электронных устройств.
  1. Аналоги разрабатываемого приложения

В данной курсовой работе я сравнил несколько подобных приложений для анализа предстоящих достоинств и недостатков.

Первое приложение «Samsung Calculator»

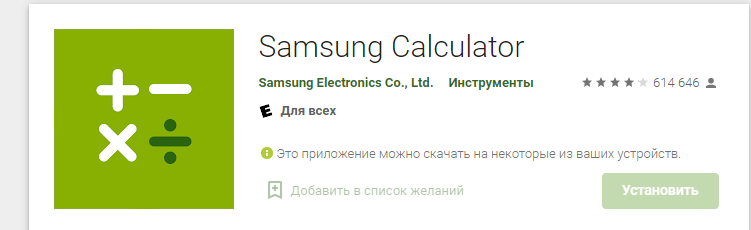


Рисунок 1 – Иконка приложения Samsung Calculator

Таблица 1 – Плюсы и минусы приложения Samsung Calculator

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Понятный интерфейс | Приложение содержит рекламу |
| Бесплатно | Раздражительный звук кнопок, который нельзя отключить |
| Стабильная и быстрая работа приложения |  |

Следующую приложение «Калькулятор» от компании «Google LLC»

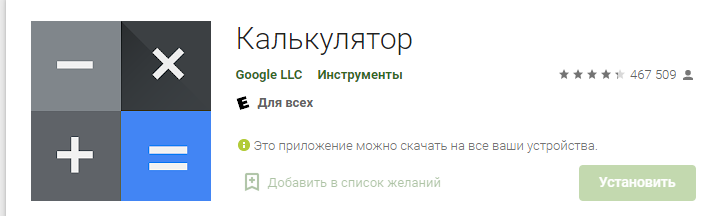


Рисунок 2 – Иконка приложения Калькулятор

Таблица 2 - Плюсы и минусы приложения Калькулятор

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Красивый дизайн | Неудобный интерфейс |
| Полное отсутствие рекламы | Кнопки вибрируют и это нельзя отключить |
| Способен выполнять сложные операции: тригонометрические, логарифмические и экспоненциальные функции. | Может выдать критическую ошибку |

Дальше у нас приложение для определения процентов: «Калькулятор с процентами»

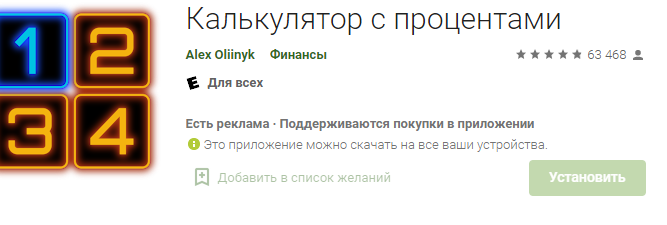


Рисунок 3 – Иконка приложения «Калькулятор с процентами»

Таблица 3 – Плюсы и минусы калькулятора с процентами

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Более точный подсчет процентов а так же их округление | Реклама |
| Удобный и красивый дизайн интерфейса |  |
| Возможность редактировать оформление |  |

* 1. Техническое задание

Программа «Калькулятор в различных система счисления», которую необходимо написать в данной работе, предназначена для решения простых арифметических задач, а так же решение арифметических задач с дробями и т.п.

Программа имеет доступ к вычислениям дробей.

Все функции достаточно понятны и востребованы в быту каждого человека.

* 1. Описание структуры приложения

Ниже представлена блок-схема алгоритма.

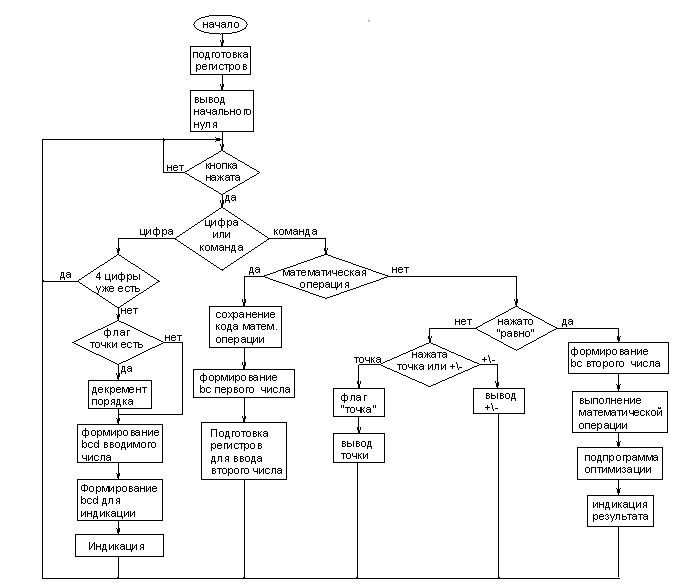


Рисунок 4 - Блок-схема алгоритма работы программы

В начале программы подготавливаются все используемые регистры, и выводится ноль на индикацию. Затем, начинается описанное выше, сканирование клавиатуры. После нажатия кнопки, определяется, что было нажато – цифра или команда. При нажатии цифры, следует проверка на переполнение водимого числа (не больше 4-х цифр) и уменьшение его порядка, если до этого была нажата точка. Затем формируются двоично-десятичные коды для преобразования числа в двоичный код, и отдельно – для индикации. После этого вводимое число появляется на индикаторе. Если была нажата математическая команда, то ее код сохраняется в специальном регистре, и формируется двоичный код первого числа, который заносится в регистр первого числа. Затем подготавливаются регистры для ввода второго числа, и микроконтроллер снова переходит к сканированию клавиатуры. При нажатии точки, устанавливается флаг «точка», при наличии которого, происходит декрементирование порядка вводимого числа с каждой новой нажатой цифрой. Если же была нажата команда +/-, то происходит инвертирование знака вводимого числа. При нажатии кнопки «равно», формируется двоичный код второго числа, который заносится в регистр второго числа. Затем считывается код нажатой математической операции, и происходит ее выполнение. После этого результат оптимизируется и выводится на индикатор.

2 Описание разработки приложения

Выбор операционной системы:

Для iOS необходимо владение Swift и Objective-C. Для Android – Kotlin или Java. Больше того, вам потребуется разная техника. Для iOS – Mac или iPhone. Для Android подойдет любой компьютер и смартфон. Существуют эмуляторы, но для постоянной проект и тестирования этот вариант не подходит.

Оба направления разработки перспективны. Однако если посмотреть на статистику, по данным Statcounter.com, Android лидирует среди всех операционных систем на мировом рынке. Он установлен на 40% всех устройств. Среди мобильных телефонов доля Android – 70%, iOS – 29%. Среди планшетов: Android – 40%, iOS – 60%. По России картина немного отличается, но тренд совпадает с общемировым. Среди смартфонов Android – 80%. На планшетах: Android – 48%, iOS – 52%. Есть региональные различия. Так, в Москве и Красноярске более популярен iOS, в Петербурге и Новосибирске – Android.

Android:

Преимущества для разработчиков:

* Для проект подойдет любой компьютер;
* Большое сообщество разработчиков;
* Чзык программирования Java может быть использован при написании не только мобильных, но и веб-приложений, игр;
* На Android работают не только приложения для смартфонов, но и для умных часов, Smart TV, VR/AR;
* Среда разработки анализирует код и может автоматически формировать некоторые его части;
* Хороший фреймворк; можно найти готовые модули под самые разные цели;

Платформу можно адаптировать под любую специализированную задачу;

* Исходный код Android открыт для всех; есть платформа Android Open Source Project, где компания публикует все данные, и разработчики тоже могут это делать;
* Публикация приложений в Google Play происходит практически мгновенно, обычно без участия модераторов – не нужно долго ждать подтверждения;
* Небольшая плата за размещение приложений в Google Play ($25 один раз, при регистрации).

Недостатки:

* Многообразие устройств и их параметров создает почву для багов, неодинаковой проект приложения на разных смартфонах;
* Многообразие версий операционной системы: в отличие от iOS, у Android остаются актуальными и могут работать без обновлений устаревшие версии;
* Не все обучающие материалы и ответы на форумах актуальны: система существует давно, нюансы разработки меняются;
* Приложение и аккаунт могут быть заблокированы без объяснения причин, на жалобы в техподдержку отвечает бот.

iOS:

Преимущества для разработчиков:

Задача состоит в том, чтобы с наибольшей точностью автоматизировать действия, которые будут выполняться вручную. Будем использовать приложения и устройства с Android. Для каждого приложения и каждого устройства нужно выполнить следующие действия:

Установить приложение на устройство

1. Запустить приложение
2. Протестировать приложение, используя выбранный метод
3. Удалить приложение
4. Сбросить устройство в исходное состояние
5. На каждом этапе нужно собирать и анализировать данные (журналы и снимки экрана).

* Строго ограниченное количество версий смартфона, размеров экрана и самой операционной системы;
* Все изменения в ОС прозрачны: Apple ежегодно выпускает гайдлайны по разработке приложений для iOS и объясняет, как заменить элементы кода на более производительные;
* Приложений и разработчиков меньше, конкуренция ниже.

Недостатки:

* Для проект необходима дорогостоящая техника;
* Чтобы загружать приложения в AppStore, нужно ежегодно продлевать аккаунт разработчика за $99;
* Публикация приложения может занять больше недели, потому что модерация производится вручную.

Вывод: Андройд лучше апле сосёт.

Выбор среды разработки:

1. Eclipse

Eclipse – это бесплатная среда разработки от некоммерческой организации Eclipse Foundation.

Благодаря активному развитию, а также поддержке со стороны компании и сторонних разработчиков, на данный момент у этой IDE имеются следующие преимущества:

* Официальная русификация интерфейса и документации;
* Отличная производительность на слабых машинах;
* Большое число дополнений (например, для проект с сервером, базой данных и т. д.);
* Возможность подключения модулей (об этом было сказано выше);
* Возможность групповой разработки.

Eclipse была очень популярна несколько лет назад и считалась монополистом на рынке IDE для Android. Однако в связи с выходом Android Studio, в 2014 г. Google перестала поддерживать Eclipse как основную среду для разработки приложений под Android.

* 1. Обоснование средств разработки

Как правило, современные средства создания Java-приложений поддерживают разработку приложений для нескольких различных платформ, а зачастую и сами существуют в версиях для различных платформ. Обычно в их число входят Windows, Linux, Solaris, а иногда и другие операционные системы.

Среда Java - это сложный механизм, обеспечивающий высокоэффективную работу программиста. Визуально она реализуется несколькими одновременно раскрытыми на экране окнами. Окна могут перемещаться по экрану, частично или полностью перекрывая друг друга, что обычно вызывает у пользователя, привыкшего к относительной «строгости» среды текстового процессора Word или табличного процессора Excel, ощущение некоторого дискомфорта.

* 1. Разработка интерфейса

Основные составные части Java

* Дизайнер форм.
* Окно редактора исходного текста.
* Палитра компонентов.
* Инспектор объектов.

Дизайнер форм в Java прост в использовании, поэтому создать визуальный интерфейс очень просто. Дизайнер форм первоначально состоит из одного пустого окна, которое заполняется разными объектами, расположенными на Палитре компонентов. Дизайнер форм изображен на рисунке 5.

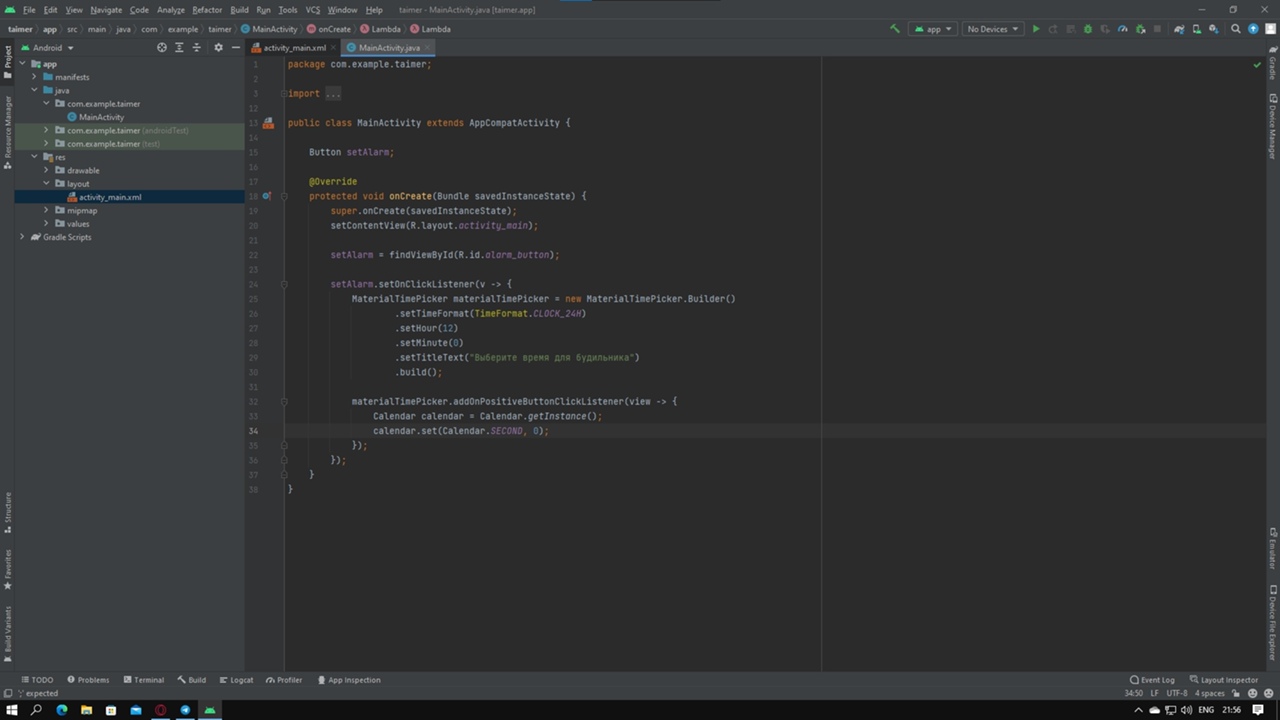


Рисунок 5 - Дизайнер форм в Java

Редактор кода - это, область в которой можно прописывать и редактировать код программы.

В приложении калькулятора есть четыре основных элемента:

[RelativeLayout](https://developer.android.com/reference/android/widget/RelativeLayout.html) - определяет, как другие элементы будут укладываться или отображаться на экране. **RelativeLayout** используется для позиционирования дочерних элементов по отношению друг к другу или к самим себе.

[TextView](https://developer.android.com/reference/android/widget/TextView.html) - элемент используется для отображения текста. Пользователи не должны взаимодействовать с этим элементом. С помощью **TextView** отображается результат вычислений.

[EditText](https://developer.android.com/reference/android/widget/EditText.html) - похож на элемент TextView, с той лишь разницей, что пользователи могут взаимодействовать с ним и редактировать текст. Но поскольку калькулятор допускает только фиксированный набор вводимых данных, мы устанавливаем для него статус «*не редактируемый*». Когда пользователь нажимает на цифры, мы выводим их в **EditText**.

[Button](https://developer.android.com/reference/android/widget/Button.html) - реагирует на клики пользователя. При создании простого приложения для Андроид мы используем кнопки для цифр и операторов действий в калькуляторе.

* 1. Разработка логики работы приложения. Схема взаимодействия компонентов проекта.

Назначение программы – Решение простых и дробных арифметических задач

В рамках курсового проекта я создал макет пользовательского интерфейса приложения, теперь мне необходимо разработать схему окон приложения и служб данного приложения, для представления о выполняемых приложением функциях и происходящих при этом процессах.

Для работы программы необходимо:

* процессор с частотой 100 МГц и не менее 400МГц
* операционная система Windows
* манипулятор мышь и клавиатура
* объем дискового пространства 5 Мб
* оперативная память 64 Мб и более.

Для мобильного приложения активности (деятельности) определяют функциональные возможности любого приложения на платформе Android.

Существует четыре типа компонентов, применяемых в среде Android:

* Activity - деятельность;
* Service - служба;
* Content Provider – контент-провайдер;
* Broadcast Receiver – приёмник широковещательных намерений (асинхронных сообщений, активирующих деятельности, службы и приемники широковещательных намерений.

Намерение – это объект класса Intent, представляющий собой содержание сообщения.

2.4 Описание переменных, компонентов, классов и подпрограмм

Один из используемых компонентов в коде данного разрабатываемого приложения является Intent

Intent - это механизм для описания одной операции - выбрать фотографию, отправить письмо, сделать звонок, запустить браузер и перейти по указанному адресу. В Android-приложениях многие операции работают через намерения.

Но это не единственный вариант использования намерения. Также можно использовать для объявления о запуске активности или сервиса, направленных на выполнение каких-либо действий (как правило, речь о работе с определенной частью данных) или для передачи уведомлений о том, что произошло некое событие (или действие).

Намерения могут применяться для трансляции сообщений по системе. Любое приложение способно зарегистрировать широковещательный приёмник и отслеживать эти намерения с возможностью на них реагировать. Это позволяет создавать приложения, использующие событийную модель, в основе которой лежат внутренние, системные или сторонние события, передаваемые внешними программами.

Явные намерения:

Intent intent = new Intent(HelloWorld.this, AboutActivity.class);

startActivity(intent);

New - создание объекта который присваивается определённой переменной

Пример:

Second second = **new** Second();

В правой части с помощью ключевого слова new создается объект от класса Second. Созданный объект присваивается переменной second, чей тип указывается перед именем переменной. В случае наличия наследственных связей тип переменной может не совпадать с типом создаваемого объекта.

Class - шаблон или описание объекта,  а объект представляет экземпляр этого класса.

Класс определяется с помощью ключевого слова **сlass**:

class Person{

}

В данном случае класс называется Person. После названия класса идут фигурные скобки, между которыми помещается тело класса - то есть его поля и методы.

Для хранения состояния объекта в классе применяются поля или переменные класса. Для определения поведения объекта в классе применяются методы. Например, класс Person, который представляет человека, мог бы иметь следующее определение:

class Person{

    String name;       // имя

    int age;           // возраст

    void displayInfo(){

        System.out.printf("Name: %s \tAge: %d\n", name, age);

    }

}

В классе Person определены два поля: name представляет имя человека, а age - его возраст. И также определен метод displayInfo, который ничего не возвращает и просто выводит эти данные на консоль.

Private - закрытый класс или член класса, противоположность модификатору public. Закрытый класс или член класса доступен только из кода в том же классе.

Protected - такой класс или член класса доступен из любого места в текущем классе или пакете или в производных классах, даже если они находятся в других пакетах

Модификатор по умолчанию - Отсутствие модификатора у поля или метода класса предполагает применение к нему модификатора по умолчанию. Такие поля или методы видны всем классам в текущем пакете.

Рассмотрим модификаторы доступа на примере следующей программы:

public class Program{

    public static void main(String[] args) {

        Person kate = new Person("Kate", 32, "Baker Street", "+12334567");

        kate.displayName();    // норм, метод public

        kate.displayAge();     // норм, метод имеет модификатор по умолчанию

        kate.displayPhone();   // норм, метод protected

        //kate.displayAddress(); // ! Ошибка, метод private

        System.out.println(kate.name);     // норм, модификатор по умолчанию

        System.out.println(kate.address);  // норм, модификатор public

        System.out.println(kate.age);      // норм, модификатор protected

        //System.out.println(kate.phone);  // ! Ошибка, модификатор private

    }

}

class Person{

    String name;

    protected int age;

    public String address;

    private String phone;

    public Person(String name, int age, String address, String phone){

        this.name = name;

        this.age = age;

        this.address = address;

        this.phone = phone;

    }

    public void displayName(){

        System.out.printf("Name: %s \n", name);

    }

    void displayAge(){

        System.out.printf("Age: %d \n", age);

    }

    private void displayAddress(){

        System.out.printf("Address: %s \n", address);

    }

    protected void displayPhone(){

        System.out.printf("Phone: %s \n", phone);

    }}

В данном случае оба класса расположены в одном пакете - пакете по умолчанию, поэтому в классе Program мы можем использовать все методы и переменные класса Person, которые имеют модификатор по умолчанию, public и protected. А поля и методы с модификатором private в классе Program не будут доступны.

Если бы класс Program располагался бы в другом пакете, то ему были бы доступны только поля и методы с модификатором public.

Модификатор доступа должен предшествовать остальной части определения переменной или метода.

Return - возращает методам некоторые их значения

После оператора return указывается возвращаемое значение, которое является результатом метода. Это может быть литеральное значение, значение переменной или какого-то сложного выражения.

Пример:

public class Program{

    public static void main (String args[]){

        int x = sum(1, 2, 3);

        int y = sum(1, 4, 9);

        System.out.println(x); // 6

        System.out.println(y); // 14

    }

    static int sum(int a, int b, int c){

        return a + b + c;

    }

}

В методе в качестве типа возвращаемого значения вместо void используется любой другой тип. В данном случае метод sum возвращает значение типа int, поэтому этот тип указывается перед названием метода. Причем если в качестве возвращаемого типа для метода определен любой другой, отличный от void, то метод обязательно должен использовать оператор **return** для возвращения значения.

При этом возвращаемое значение всегда должно иметь тот же тип, что значится в определении функции. И если функция возвращает значение типа int, то после оператора return стоит целочисленное значение, которое является объектом типа int. Как в данном случае это сумма значений параметров метода.

Extends - это ключевое слово, предназначенное для расширения реализации какого-то существующего класса. Создается новый класс на основе существующего, и этот новый класс расширяет (extends) возможности старого.

Код кнопок в программе:

stpower2nd.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 try{  
 ValueOne = Float.*parseFloat*(txtview.getText() + "");  
 istpower = true;  
 txtview.setText(null);  
 }  
 catch (Exception e) {  
 Toast t = Toast.makeText(MainActivity.this, "Syntax ERROR", Toast.LENGTH\_LONG);  
 t.setGravity(Gravity.CENTER, 0, 0);  
 t.show();  
 }  
 }  
 });  
 epowerbtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 try {  
 Double n = Double.*parseDouble*(txtview.getText().toString());  
 Double exp = (Double) Math.*pow*(2.718281828, n);  
 txtview.setText(exp+"");  
 }  
 catch (Exception e){  
 Toast t = Toast.makeText(MainActivity.this, "Syntax ERROR", Toast.LENGTH\_LONG);  
 t.setGravity(Gravity.CENTER, 0, 0);  
 t.show();  
 }  
 }  
 });  
 modulosbutton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 try {  
 ValueOne = Float.*parseFloat*(txtview.getText() + "");  
 Reminder = true;  
 txtview.setText(null);  
  
// a = Double.parseDouble(txtbtn.getText().toString());  
// Double result = (Double) Math.cos(Math.toRadians(a));  
// txtbtn.setText(result + "");  
 }  
 catch (Exception e) {  
 Toast t = Toast.makeText(MainActivity.this, "Syntax ERROR", Toast.LENGTH\_LONG);  
 t.setGravity(Gravity.CENTER, 0, 0);  
 t.show();  
 }  
 }  
 });  
 deletebutton.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view) {  
 try {  
 String str;  
 str = txtview.getText().toString();  
 str = str.substring(0, str.length() - 1);  
 txtview.setText(str);  
 }  
 catch (Exception e){}  
 }  
 });

Так же в моем коде используются следующие библиотеки:  
android.os.Bundle;  
android.view.Gravity;  
android.view.View;  
android.widget.Button;  
android.widget.TextView;  
android.widget.Toast;

android.os.Bundle

Класс Bundle (с английского - сверток) по сути представляет собой враппер (оболочку) над коллекцией ArrayMap для создания более комфортного в работе контейнера для элементов разных типов, в котором можно разместить любые Parcelabe-объекты и примитивные типы. Класс является потокобезопасным и может использоваться для передачи значений между разными потоками. Доступ к элементам этой коллекции осуществляется, как и в ArrayMap, по парам ключ-значение. Дополнительно класс предоставляет методы по размещению и извлечению типизированных данных и некоторые другие возможности.

android.view.Gravity;

Атрибут **gravity** задает позиционирование содержимого внутри визуального элемента.

Метод View.forceLayout() инвалидирует закэшированные размеры view, чем провоцирует измерение, при вызове requestLayout() на родительском ViewGroup.

1. Тестирование и установка приложения

В ходе тестирования было выявлено что приложение работает не на всех платформах

Заключение

В этой курсовой работе я проанализировал популярные программы и создал свой аналог

Создание данного приложения не несло за собой цели конкурировать с местными приложениями, однако это может послужить хорошим началом в моей IT карьере

Так же создавая данную программу я использовал множество разных классов методов и компонентов, создав данную программу я изучил и узнал довольно много нового о создании программ и о языке java в целом, это очень ценный опыт для начинающего программиста.

Списки литературы используемые в работе:

1.http://developer.alexanderklimov.ru/android/theory/activity\_methods.php

2.https://metanit.com/java/android/2.1.php

3.https://ru.stackoverflow.com/questions/623829/

4.https://www.fandroid.info/context-kontekst-v-android-chto-eto-kak-poluchit-i-zachem-ispolzovat/

5.https://habr.com/ru/company/otus/blog/560492/

6.https://coderlessons.com/articles/mobilnaia-razrabotka-articles/rukovodstvo-po-android-alarmmanager

7.http://study-java.ru/uroki-java/urok-6-sintaksis-java-peremennye-i-tipy-dannyh/

8.http://microsin.net/programming/android/java-extends-implements-interface-override.html#:~:text=extends%20это%20ключевое%20слово%2C%20предназначенное,для%20реализации%20интерфейса%20(interface).

9.https://www.frolov-lib.ru/programming/javasamples/vol3/vol3\_1/index.html

10.https://metanit.com/java/tutorial/2.17.php

11.https://younglinux.info/java/objects

12.https://vertex-academy.com/tutorials/ru/sozdanie-peremennyx-i-tipy-peremenny/

13.http://developer.alexanderklimov.ru/android/java/set.php

14.http://proglang.su/java/modifiers#:~:text=Модификатор%20private%20—%20методы%2C%20переменные%20и,интерфейсы%20не%20могут%20быть%20private.

15.https://spec-zone.ru/RU/Java/Docs/7/api/java/text/SimpleDateFormat.html

## Приложения

Приложение А. Листинг кода

EnemySpaceShip

package com.sandipbhattacharya.spaceshooter;

import android.content.Context;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import java.util.Random;

public class EnemySpaceship {

Context context;

Bitmap enemySpaceship;

int ex, ey;

int enemyVelocity;

Random random;

public EnemySpaceship(Context context) {

this.context = context;

enemySpaceship = BitmapFactory.decodeResource(context.getResources(), R.drawable.rocket2);

random = new Random();

ex = 200 + random.nextInt(400);

ey = 0;

enemyVelocity = 14 + random.nextInt(10);

}

public Bitmap getEnemySpaceship(){

return enemySpaceship;

}

int getEnemySpaceshipWidth(){

return enemySpaceship.getWidth();

}

int getEnemySpaceshipHeight(){

return enemySpaceship.getHeight();

}

}

Explosion

package com.sandipbhattacharya.spaceshooter;

import android.content.Context;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

public class Explosion {

Bitmap explosion[] = new Bitmap[9];

int explosionFrame;

int eX, eY;

public Explosion(Context context, int eX, int eY) {

explosion[0] = BitmapFactory.decodeResource(context.getResources(),

R.drawable.explosion0);

explosion[1] = BitmapFactory.decodeResource(context.getResources(),

R.drawable.explosion1);

explosion[2] = BitmapFactory.decodeResource(context.getResources(),

R.drawable.explosion2);

explosion[3] = BitmapFactory.decodeResource(context.getResources(),

R.drawable.explosion3);

explosion[4] = BitmapFactory.decodeResource(context.getResources(),

R.drawable.explosion4);

explosion[5] = BitmapFactory.decodeResource(context.getResources(),

R.drawable.explosion5);

explosion[6] = BitmapFactory.decodeResource(context.getResources(),

R.drawable.explosion6);

explosion[7] = BitmapFactory.decodeResource(context.getResources(),

R.drawable.explosion7);

explosion[8] = BitmapFactory.decodeResource(context.getResources(),

R.drawable.explosion8);

explosionFrame = 0;

this.eX = eX;

this.eY = eY;

}

public Bitmap getExplosion(int explosionFrame){

return explosion[explosionFrame];

}

}

public class MainActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(new SpaceShooter(this));

}

@Override

protected void onDestroy() {

super.onDestroy();

}

@Override

protected void onPause() {

super.onPause();

}

}

OurSpace

package com.sandipbhattacharya.spaceshooter;

import android.content.Context;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import java.util.Random;

public class OurSpaceship {

Context context;

Bitmap ourSpaceship;

int ox, oy;

Random random;

public OurSpaceship(Context context) {

return shot.getWidth();

}

public int getShotHeight() {

return shot.getHeight();

}

}

SpaceShooter

import java.util.ArrayList;

import java.util.Random;

public class SpaceShooter extends View {

Context context;

Bitmap background, lifeImage;

Handler handler;

long UPDATE\_MILLIS = 30;

static int screenWidth, screenHeight;

int points = 0;

int life = 3;

Paint scorePaint;

int TEXT\_SIZE = 80;

boolean paused = false;

OurSpaceship ourSpaceship;

EnemySpaceship enemySpaceship;

Random random;

ArrayList<Shot> enemyShots, ourShots;

Explosion explosion;

ArrayList<Explosion> explosions;

boolean enemyShotAction = false;

final Runnable runnable = new Runnable() {

@Override

public void run() {

invalidate();

}

};

public SpaceShooter(Context context) {

super(context);

this.context = context;

Display display = ((Activity) getContext()).getWindowManager().getDefaultDisplay();

Point size = new Point();

display.getSize(size);

screenWidth = size.x;

screenHeight = size.y;

random = new Random();

enemyShots = new ArrayList<>();

ourShots = new ArrayList<>();

explosions = new ArrayList<>();

ourSpaceship = new OurSpaceship(context);

enemySpaceship = new EnemySpaceship(context);

handler = new Handler();

enemySpaceship.enemyVelocity \*= -1;

}

if(enemyShotAction == false){

if(enemySpaceship.ex >= 200 + random.nextInt(400)){

Shot enemyShot = new Shot(context, enemySpaceship.ex + enemySpaceship.getEnemySpaceshipWidth() / 2, enemySpaceship.ey );

enemyShots.add(enemyShot);

enemyShotAction = true;

}

if(enemySpaceship.ex >= 400 + random.nextInt(800)){

Shot enemyShot = new Shot(context, enemySpaceship.ex + enemySpaceship.getEnemySpaceshipWidth() / 2, enemySpaceship.ey );

enemyShots.add(enemyShot);

enemyShotAction = true;

}

else{

Shot enemyShot = new Shot(context, enemySpaceship.ex + enemySpaceship.getEnemySpaceshipWidth() / 2, enemySpaceship.ey );

enemyShots.add(enemyShot);

enemyShotAction = true;

}

}

canvas.drawBitmap(enemySpaceship.getEnemySpaceship(), enemySpaceship.ex, enemySpaceship.ey, null);

if(ourSpaceship.ox > screenWidth - ourSpaceship.getOurSpaceshipWidth()){

ourSpaceship.ox = screenWidth - ourSpaceship.getOurSpaceshipWidth();

}else if(ourSpaceship.ox < 0){

ourSpaceship.ox = 0;

}

canvas.drawBitmap(ourSpaceship.getOurSpaceship(), ourSpaceship.ox, ourSpaceship.oy, null);

for

for(int i=0; i < ourShots.size(); i++){

ourShots.get(i).shy -= 15;

canvas.drawBitmap(ourShots.get(i).getShot(), ourShots.get(i).shx, ourShots.get(i).shy, null);

if((ourShots.get(i).shx >= enemySpaceship.ex)

&& ourShots.get(i).shx <= enemySpaceship.ex + enemySpaceship.getEnemySpaceshipWidth()

&& ourShots.get(i).shy <= enemySpaceship.getEnemySpaceshipWidth()

&& ourShots.get(i).shy >= enemySpaceship.ey){

points++;

ourShots.remove(i);

explosion = new Explosion(context, enemySpaceship.ex, enemySpaceship.ey);

explosions.add(explosion);

}else if(ourShots.get(i).shy <=0){

ourShots.remove(i);

}

}

for(int i=0; i < explosions.size(); i++){

canvas.drawBitmap(explosions.get(i).getExplosion(explosions.get(i).explosionFrame), explosions.get(i).eX, explosions.get(i).eY, null);

explosions.get(i).explosionFrame++;

if(explosions.get(i).explosionFrame > 8){

explosions.remove(i);

}

}

if(!paused)

handler.postDelayed(runnable, UPDATE\_MILLIS);

}

@Override

public boolean onTouchEvent(MotionEvent event) {

int touchX = (int)event.getX();

if(event.getAction() == MotionEvent.ACTION\_UP){

if(ourShots.size() < 1){

Shot

onTouchEvent() tells Android system that you already handled

return true;

}

}

StartUp

package com.sandipbhattacharya.spaceshooter;

import android.content.Intent;

import android.os.Bundle;

import android.view.View;

import androidx.annotation.Nullable;

import androidx.appcompat.app.AppCompatActivity;

public class StartUp extends AppCompatActivity {

@Override

public void onCreate(@Nullable Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.startup);

}

public void startGame(View view) {

startActivity(new Intent(this, MainActivity.class));

finish();

}

}

Приложение Б. Руководство пользователя.

Руководство пользователя мобильного приложения для устройств на ОС Android

1. Требования к аппаратуре и программному обеспечению

Для устройств на базе ОС Android:

Android — версия 5.0 и старше; сенсорный экран.

2. Установка, обновление, удаление мобильного приложения

Установка мобильного приложения выполняется из магазина приложений Google Play (https://play.google.com/store).

Обновление мобильного приложения выполняется средствами магазина приложений.

Для удаления мобильного приложения необходимо на мобильном устройстве запустить приложение Google Play и в нем удалить мобильное приложение.

3. Запуск мобильного приложения

Для запуска приложения нужно найти его в списке и нажать на иконку. При этом будет открыто основное окно мобильного приложения.

4. Интерфейс системы

Интерфейс мобильной платформы ориентирован на то, что в один момент времени на экране отображается одна форма. Под нее освобождается максимум свободного места на экране. Первой видимой формой является главный экран. Команды, открывающие другие экраны, находятся в главном меню приложения. Также это меню содержит команду возврата на рабочий стол, команды отображения информации о программе и перехода к списку приложений. Вызов главного меню приложения осуществляется с помощью системной кнопки вызова меню.

1. Выход из приложения

Для выхода из приложения достаточно свайпнуть по экрану согласно системе или нажать кнопку назад на нижней панели управления.

Рецензия

В ходе тестирования данного приложения становится ясно, что оно требует внимательности и реакции. Интерфейс очень приятен глазу, присутствуют мягкие цвета и оттенки.

Так же присутствуют и недостатки: нет сохранения рекорда. Нет анимации перехода на новые уровни или усложнения игры, что выглядит как некачественно выполненная работа.

В ходе тестирования можно сказать, что для данного приложения нужны правки и доработки. При исправлении этих недостатков приложение приобретёт новые вид и впечатления, а за ними и большую заинтересованность пользователей к игре, так как чем качественней продукт, тем больше он востребован.

Ссылка на github: https://github.com/Matsuda/Kalculator.git