федеральное государственное автономное образовательное учреждение высшего образования

Санкт-Петербургский национальный исследовательский УНИВЕРСИТЕТ

информационных технологий, механики и оптики

Факультет \_\_ПИиКТ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(название факультета)

Кафедра \_\_КПиД\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(название кафедры)

Направление подготовки (специальность) Веб-технологии

**О Т Ч Е Т**

**о производственной практике**

(наименование практики)

**Тема задания:** Создание приложений с использованием языка программирования Dart

**Студент Бабарицкий Павел Александрович S4107\_**

(Фамилия И.О.) номер группы

**Руководитель практики от организации:** ФИО Шуклин Д.А. (практика на кафедре) зав. кафедрой КПиД, к.п.н., доцент

(Фамилия И.О., должность и место работы)

**Ответственный за практику от университета:** Шуклин Дмитрий Анатольевич, зав. кафедрой КПиД, к.п.н., доцент

**Практика пройдена с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Подписи членов комиссии**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (**зав. кафедрой

к.п.н., доцент

Шуклин Д.А**)**

(подпись)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (** зам. декана "Академии ЛИМТУ" к.т.н.,   
доцент Сокуренко Ю.А.**)**

(подпись)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_(** доц. кафедры

КПиД, к.т.н.

Государев И.Б.**)**

(подпись)

**Санкт-Петербург**

**2018**

**ОГЛАВЛЕНИЕ**

|  |  |
| --- | --- |
| Введение . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 3 |
| Глава 1 Обзор программного средства Dart . . . . . . . . . . . . . . . . . . . . . . . . | 4 |
| Глава 2 Описание выполнения практической работы. . . . . . . . . . . . . . . . . | 8 |
| Заключение . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 18 |
| Библиографический список. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . | 19 |

**ВВЕДЕНИЕ**

Для прохождения практики выбрана кафедра КПиД факультета ПИиКТ университета ИТМО. Созданные условия для прохождения практики максимально приближены к условиям работы по будущей профессиональной деятельности, а задания позволяют получить необходимые практические знания.

Технологией для изучения был выбран язык - Dart. В качестве практического задания было необходимо исследовать теоретическую информацию, изучить данный язык разработки для создания веб-приложений, консольных приложений и приложений для мобильных устройств и создать обучающий скринкаст с логом освоения этого средства разработки.

Так же следует отметить что выполняемые в ходе практики работы позволили получить навыки в использование современных технологий и методов разработки сайтов.

**ГЛАВА 1 ОБЗОР ПРОГРАММНОГО СРЕДСТВА DART**

Dart — язык программирования, созданный Google. Dart позиционируется в качестве замены/альтернативы JavaScript. Один из разработчиков языка Марк Миллер (Mark S. Miller) написал, что JavaScript «имеет фундаментальные изъяны» («Javascript has fundamental flaws…»), которые невозможно исправить [1]. Поэтому и был создан Dart.

Первая общедоступная информация об этом языке программирования появилась 12 сентября 2011 года на конференции разработчиков Goto. 10 октября 2011 была проведена официальная презентация языка Google Dart.

* Стоит отметить, что знание TypeScript совместно с JavaScript практически полностью покрывает возможности Dart. С другой стороны, потенциал Dart куда выше и его стоит рассматривать скорее, как этап развития, а не как конкурента JavaScript [2].
* Главные особенности — это наличие наследования и строгой типизации.

Задачи, поставленные перед разработчиками языка:

* Создать структурированный и в то же время гибкий язык для веб-программирования
* Сделать язык похожим на существующие для упрощения обучения
* Высокая производительность получаемых программ как в браузерах, так и в иных окружениях, начиная от смартфонов и заканчивая серверами.

Изначально было предложено два способа исполнения Dart-программ: с использованием виртуальной машины (в поддерживающих язык браузерах) или с промежуточной трансляцией в javascript (универсальный)

15 ноября 2013 года Google выпустили первую стабильную версию своего языка программирования — Dart SDK 1.0.

4 июля 2014 года ECMA одобрили первую редакцию стандарта языка, стандарт получил название ECMA-408.

Приложения, создаваемые с использованием языка Dart:

1. Web приложения (Приложение на Dart переводится в JS)
2. Dart VM - Серверное программирование (Создание консольных приложений)
3. Flutter - Для мобильных устройств (Код компилируется в код для конкретного устройства)

Особенности языка Dart:

1. Код Dart совместим со всеми основными браузерами, без необходимости использования браузерами Dart. Благодаря оптимизации скомпилированного вывода JavaScript, чтобы избежать дорогостоящих проверок и операций, код, написанный в Dart, в некоторых случаях может работать быстрее, чем эквивалентный код, написанный вручную на JavaScript.

Комплект разработки программного обеспечения Dart (SDK) поставляется с версией браузера Chromium, модифицированной для включения виртуальной машины Dart (VM). Этот браузер может напрямую запускать код Dart без компиляции в JavaScript. Он предназначен как инструмент для разработки приложений, написанных на этом языке, а не как универсальный веб-браузер. Первоначально планировалось включить поддержку Dart прямо в Chrome, но они были отменены. Веб-браузер Dartium будет прекращен с выпуском Dart 2.0, благодаря новому компилятору разработки dartdevc.

2. Dart SDK поставляется с автономной Dart VM, позволяя запускать код в среде интерфейса командной строки. Dart поставляется с полной стандартной библиотекой, позволяющей пользователям писать полностью работающие системные приложения, такие как пользовательские веб-серверы.

3. Приложения, созданные с помощью Flutter, SDK для мобильных приложений, созданного с помощью Dart, развертываются в магазинах приложений как AOT(Ahead-of-Time)-скомпилированный код Dart. (AOT компилятор — это вид транслятора, который использует метод компиляции перед исполнением)

Flutter — фреймворк для кроссплатформенной разработки мобильных приложений.

Flutter сделан на С и С++, реализует свой 2D движок для рендеринга (WebView не используется). Чем-то похож на React, разработка ведется на языке Dart. Код пишется один раз, а при сборке компилируется в нативный для каждой платформы.

У Flutter есть списком преимуществ:

* кроссплатформенность: тратим меньше времени на разработку за счет переиспользования кода;
* быстрота работы: AOT компиляция в нативный код;
* набор компонентов: есть много типовых виджетов  в стилях Material Design и Cupertino;
* hot reload: горячая перезагрузка (можем менять код и видеть результат в уже открытом приложении на мобильном устройстве) [3].

Разработчики языка придерживаться следующих принципов:

* язык не может уходить слишком далеко от знакомых вещей, если он хочет оставаться понятным миллионам разработчиков;
* язык должен поддерживать реактивный стиль программирования;
* синтаксис должен быть четким и кратким, инструментарий — простым, а цикл разработки — почти мгновенным;
* производительность запуска и выполнения должны быть отличными и предсказуемыми даже на медленных мобильных устройствах;
* разработчики клиентских приложений должны сегодня думать о трех платформах: iOS, Android и Web. Язык должен хорошо работать на всех из них [4].

**ГЛАВА 2 ОПИСАНИЕ ВЫПОЛНЕНИЯ ПРАКТИЧЕСКОЙ РАБОТЫ**

Для создания скринкаста были заранее установлены Dart, Flatter и среды для работы с ними. Краткое пояснение по их установке было включено в скринкаст, а также их установка описана далее.

Записываемый скринкаст по Dart придерживается следующего плана:

1. Установка рекомендуемых IDE, редакторов, фреймворков, то есть всего, что необходимо для работы с Dart.

2. Инструкция по использование менеджера пакетов pub

3. Создание простейшего консольного приложения с использованием Dart

4. Создание простейшего Web приложения

5. Установка и создание простейшего Flutter приложения

Каждый этап подробно описан ниже в данной главе.

**Пошаговая инструкция по выполнению работы:**

**Часть 1 Установка**

1. Все программы для работы с Dart были установлены на операционную систему Windows 10. В инсталляторах настройки были оставлены по умолчанию и был изменён только путь сохранения, там, где это было возможно.

2. Для начала необходимо скачать Git, это можно сделать с официального сайта. https://git-scm.com -> Downloads [5].

3. Теперь можно установить Dart с сайта: https://www.dartlang.org -> Install Dart -> Dart VM -> Dart SDK installer for Windows -> get dart 64 bit windows dev [6].

4. В качестве редактора рекомендую использовать Visual Studio code https://code.visualstudio.com/download - откройте её после установки и установите расширения: Dart Code 2.11.2 и Flutter snippets [7].

5. Для удобства работы установите фреймворк для быстрого создания приложений на dart.

Для этого необходимо создать путь с именем:

C:\Users\”Имя Пользователя”\AppData\Roaming\Pub\Cache\bin

Например: C:\Users\1\AppData\Roaming\Pub\Cache\bin

Необходимо прописать PATH: C:\Users\1\AppData\Roaming\Pub\Cache\bin. Для этого нужно перейти в: Система -> Дополнительные параметры системы -> Переменные среды -> Path -> Изменить -> Создать, пропишите путь к папке bin.

Откройте командную строку в C:\Users\1\Documents и выполните консольную команду

pub global activate stagehand

Выполните pub global list – чтобы увидеть, что stagehand успешно установлен.

Если есть необходимость обновить библиотеки pub, то можно использовать консольную команду

pub upgrate

На этом установка Dart Завершена.

**Часть 2 Создание консольного приложения**

1. Откройте Visual Studio Code и создаём простейшее консольное приложение:

* Файл –> Открыть папку -> Создаем папку нового проекта.
* Отрываем консоль –> терминал и вызываем команду stagehand, можно увидеть, какие команды нам доступны, выберете console-full – консольное приложение.
* Если вы хотите изменить оболочку терминала, то нажмите Ctr+Shift+P и наберите:

“Терминал: Выбрать оболочку по умолчанию” и выберите из списка.

* Вызываем команду pub get – для установления зависимостей.
* Получили проект по умолчанию, который выводит результат умножения двух чисел в консоль.

2. Рассмотрим создание проекта с нуля, чтобы разобраться из чего он должен состоять:

pubspec.yaml – файл в котором содержится мета информация и описаны подключаемы пакеты, в соответствии с ним при вызове команды pub get устанавливаются зависимости

На сайте https://pub.dartlang.org можно посмотреть какие пакеты существуют и как их подключить [8].

*Файл pubspec.yaml:*

name: prog1

version: 0.0.1

description: Hello Dart for VM

author: Pavel

depndencies:

dependencies:

test: "^0.12.33"

executables:

make\_hello: main

Папка lib в ней создайте hello\_dart\_prog.dart в котором будут подключаться свои созданные ресурсы из папки src, а в src создайте файл hello.dart с классом Hello и его методом sayHello выводящей приветствие.

*Файл hello\_dart\_prog.dart:*

export 'src/hello.dart';

*Файл hello.dart:*

class Hello{

final String text\_hello = "Hello World 123";

void sayHello(){

print(text\_hello);

}

}

Папка bin в ней main.dart – основной исполняемый файл.

*Файл main.dart*

import 'package:prog1/hello\_dart\_prog.dart';

void main(){

new Hello().sayHello();

}

Создадим каталог для тестов, папка test в ней создаем main\_test.dart.

*Файл main\_test.dart*

import 'package:test/test.dart';

import '../bin/main.dart' as maintest;

import 'package:prog1/hello\_dart\_prog.dart';

void main() {

test("Hello World 123", (){

expect(maintest.main, prints("Hello World 123\n"));

// одидаем что maintest.main напишет Hello World 123

// callback ф-ия выполняющая тест

expect(new Hello().sayHello, prints("Hello World 123"));

// Проверяем что ф-ия печатает нужный текст

});

}

Команда pub run test – запускает на исполнение созданные Unit тесты.

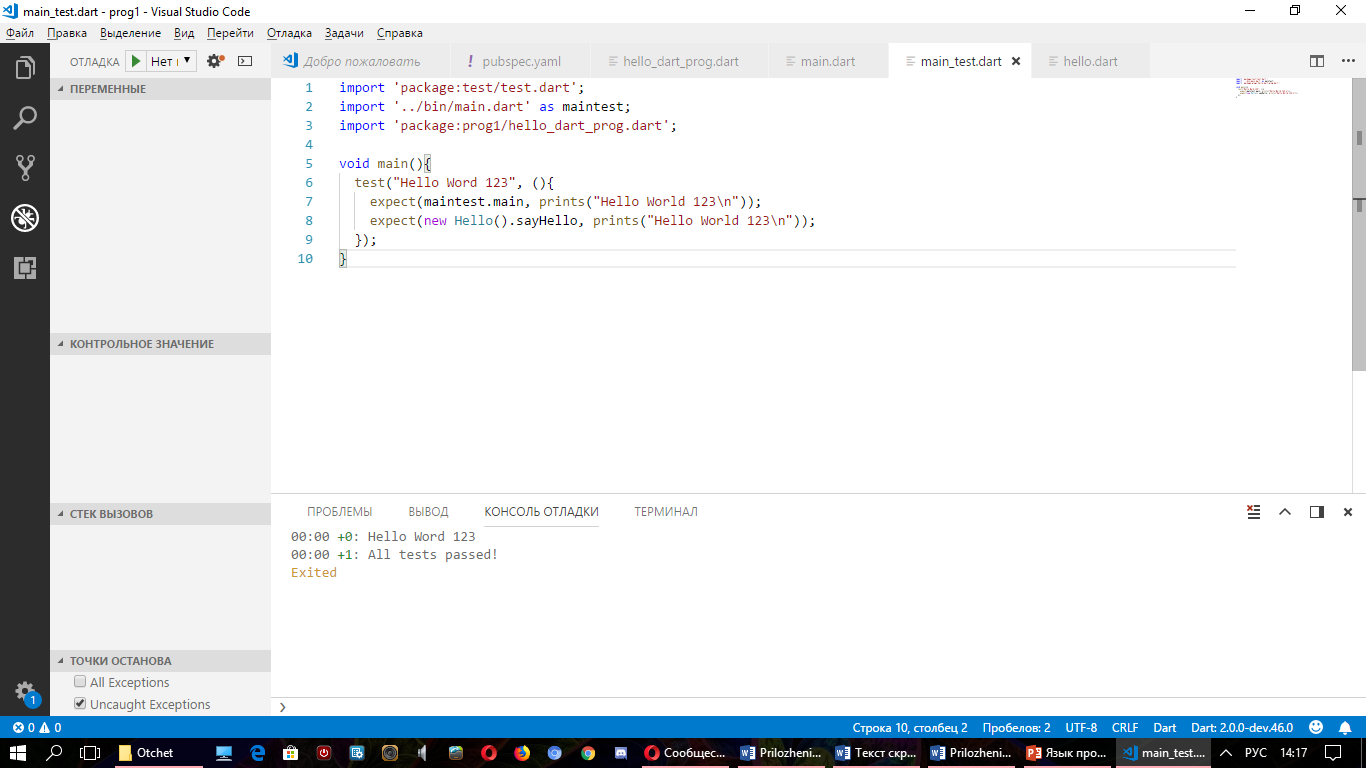


Рисунок 1. Результат выполнения созданного консольного приложения с использованием языка Dart

**Часть 3 Создание веб-приложения**

Создадим простое веб-приложение, для этого создайте и откроете папку нового проекта и в терминале вызовите команду stagehand web-simple, а затем pub get.

Файлы проекта автоматический создались в папку web, мы получили приложение, которое выводит текст 'Your Dart app is running.'

На html страницу по id="output"

В файле yaml уже прописаны подключаемые пакеты

browser: ^0.10.0 – для поддержки браузеров не работающих с Dart, он изменяет ссылку скриптов файла Dart на советующий JavaScript файл.

dart\_to\_js\_script\_rewriter: ^1.0.1

transformers:

- dart\_to\_js\_script\_rewriter – перезаписывает подключение файла main.dart на maint.dart.js

Если нужно вывести текст, например, в тег h1 то достаточно написать следующее в файле dart:

const text = 'Hello Dart Web';

Element h1 = querySelector('h1');

h1.text = text;

В html файле добавили:

<h1></h1>

Если мы хотим передать в <div id="output"></div>

Текущее время, то для этого в файл pubspec.yaml добавим пакет:

intl: "^0.15.6";

А в файл main.dart:

import 'dart:core';

import 'package:intl/intl.dart';

В результате подключили пакеты core для работы с датой и intl.dart для изменения её формата [9], [10].

Вызвали pub get чтобы подключить новые зависимости.

В файле пишем следующее

DateTime now = new DateTime.now();

DateFormat dateFormat1 = new DateFormat.yMd().add\_jms();

DateFormat dateFormat2 = new DateFormat("yyyy.MM.dd");

String formated = dateFormat2.format(now);

querySelector('#output').text = formated;

Можно задать формат даты двумя способами.

Более подробно со всеми возможностями Dart можно ознакомится в документации на официальном сайте [10].

Чтобы собрать проект и скомпилировать в javascript, выполняется команда:

pub build

которая создает build приложение.

Чтобы запустить локальный сервер выполните pub serve –-port 8083 можно указать порт, если порт по умолчанию 8080 у вас используется.

В браузере можно увидеть, что проект загружается с задержкой т.к. тратиться время на создание js файла. Если открыть index.html из папки build то он отроется без задержки. Т.к. файл ссылается на уже созданный js файл.

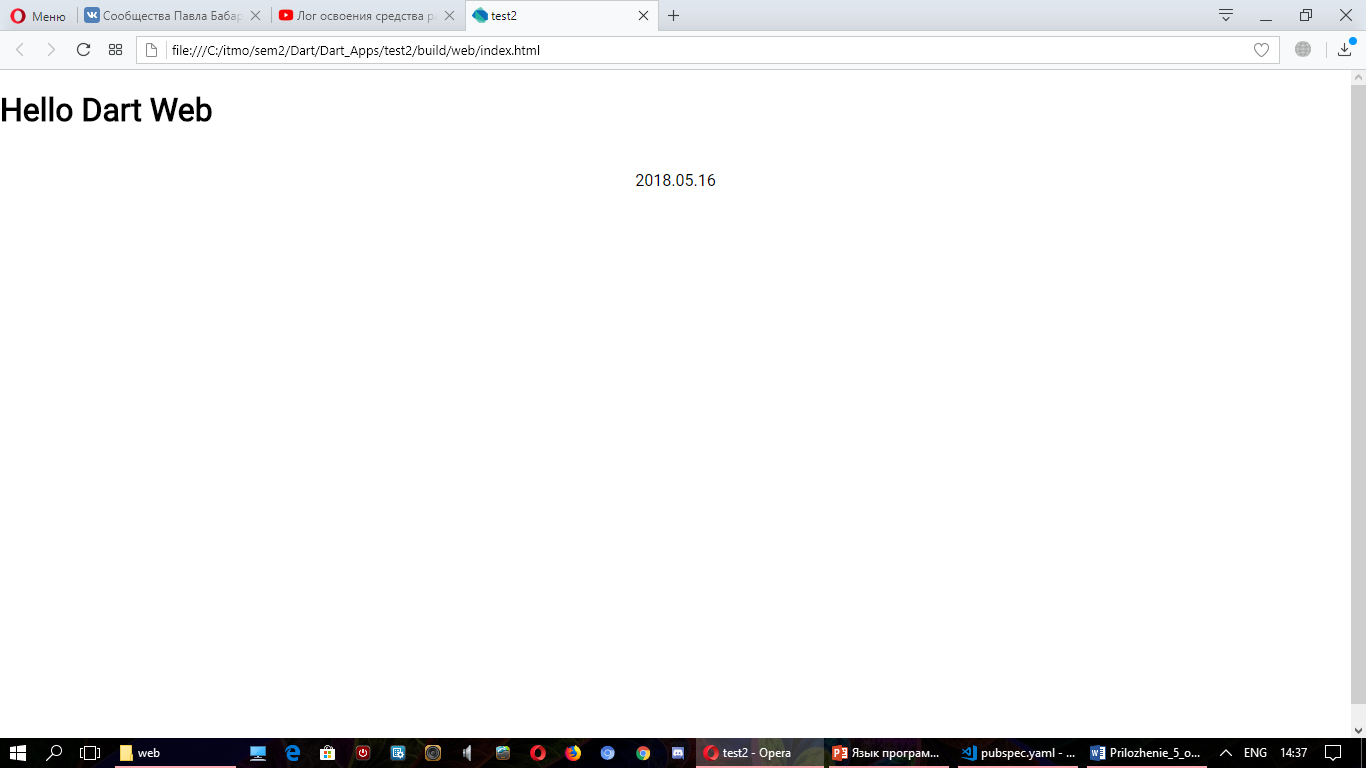


Рисунок 2. Результат созданного веб-приложения с использованием языка Dart

**Часть 4 Установка и создание мобильного приложения**

Теперь поговорим о flutter – это фреймворк для создания мобильных приложений [11].

Для его установки нужно сделать следующее:

Перейти на сайт https://flutter.io/setup-windows/ и установите zip архив или на сайте https://flutter.io/sdk-archive/ скопировав команду установить через консоль, открыв её в той папке куда вы собираетесь установить [12].

git clone -b master https://github.com/flutter/flutter.git

./flutter/bin/flutter --version

Разархивируйте файлы например в папку flutter на диске C. Теперь аналогично фреймворку stagehande нужно прописать путь в систему. Откройте параметры среды и прописшите путь, до папки bin. C:\flutter\flutter\bin\

После чего коммандой flutter doctor установите зависимости.

Далее устанавите Android Studio для возможности эмулировать Android устройства:

https://developer.android.com/studio/index.html [13].

А в качестве редактора рекомендую использовать IntelliJ IDEA Community:

https://www.jetbrains.com/idea/download/#section=windows [14].

С установкой этих программ проблем возникнуть не должно, просто оставляйте настройки по умолчанию.

Запустите Android Studio и создадайте виртуальное устройство.

Нажиммите на иконку ADV Manager – создание виртуального устройства.

Далее Create Virtual Device. Выберете модель, например, Nexius 5. Затем во вкладке образов x86 Images выбираю образ Naugat последней версии для 32-х и 64-х разрядных систем. В Graphics выберете HardWare – GLES 2.0. После чего нажимаем Finish и устройство через некоторое время будет создано и появится в списке устройств.

После этого запустите эмулятор нажав на зеленую стрелку. Когда эмулятор у вас запустится откройте консоль и выполните команду flutter doctor, для установления зависимостей. Можете пока закрыть эмулятор.

Откроем редактор IntelliJ. Когда откроете перейдите в настройки конфигурации (Edit->Settings), там выберите вкладку Plugins->Browse repositories, вбейте в поиск flutter и установите его. Также зайдите в File -> Project Structure и добавьте путь к папке с JDK.

Создайте новый проект File->New->Project…->Flutter. Во Flutter SDK path указать путь к папке, в которой у вас установлен flutter. У меня это C:\flutter\flutter.

Далее укажите имя проекта обязательно строчными буквами и путь к папке с проектом.

После создания откройте файл pubspec.yaml и нажмите в верхнем меню файла Flutter сommands, команду Flutter upgrade.

Запускаем эмулятор, после чего можно нажать на зеленую кнопку Run main.dart в IntelliJ.

В результате было создано и запущено приложение, которое создается по умолчанию. У нас есть кнопка и при нажатии на неё накапливается счетчик, и его накопление отображается на экране устройства.

Можете изменить выводимый текст в приложении, изменив строчку с этим текстом в коде приложения и затем, перезапустив приложения и нажав на кнопку отладки, вы мгновенно уведите полученный результат.

Более подробно о том, как создавать приложения для мобильных устройств на языке Dart вы также можете узнать из документации, на сайте фреймворка: https://flutter.io [15].

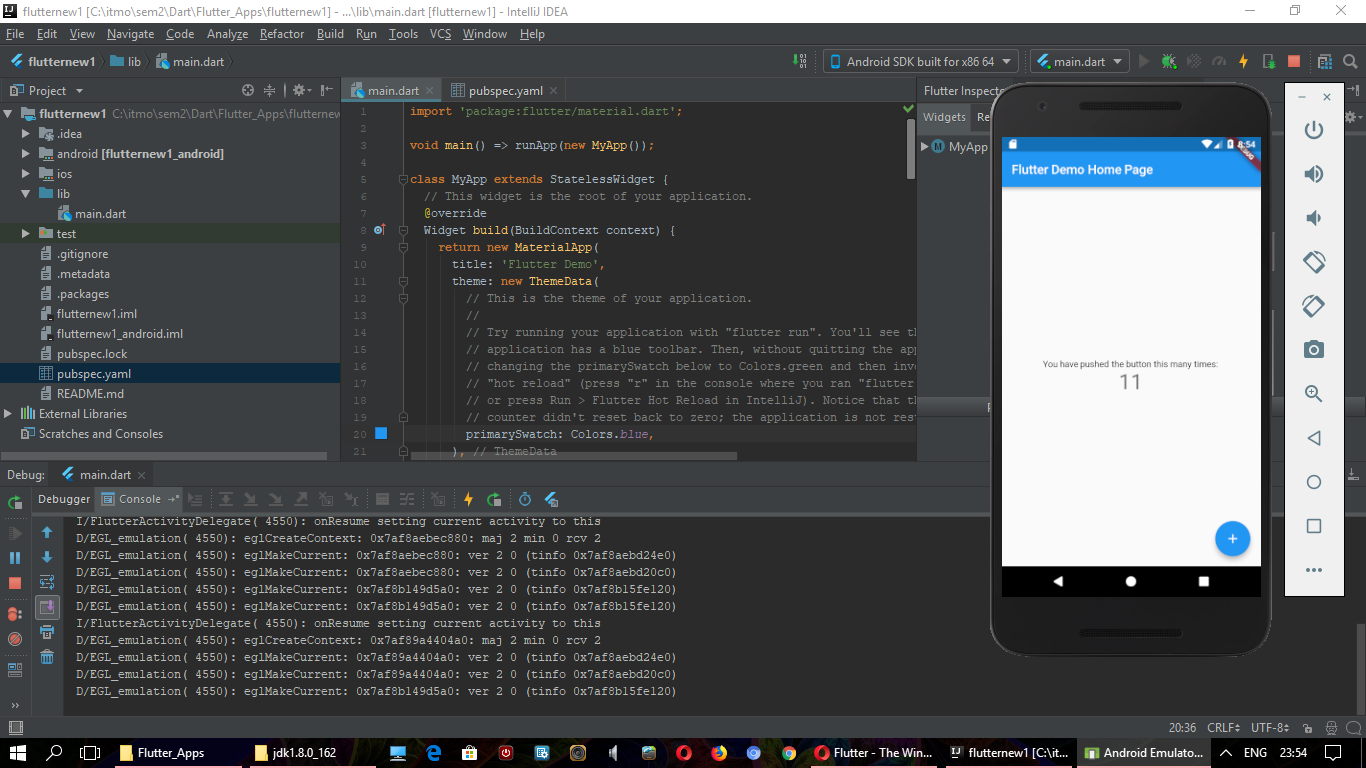


Рисунок 3. Результат выполнения созданного мобильного приложения на Flutter

**ЗАКЛЮЧЕНИЕ**

В процессе выполнения практического задания был изучен язык Dart в процессе его рассмотрения, создан скринкаст, в нём продемонстрировано создание консольных приложений, веб-приложений и мобильных приложений с использованием языка Dart. В видео был объяснён процесс установки требуемых и рекомендуемых приложений для работы с языком, а также рассмотрена работа с менеджером пакетов pub. Результаты и процесс выполняемой работы были отмечены в отчёте в виде пошаговой инструкции, раскрывающей и поясняющей работу с исследуемым языком.

**СПИСОК ЛИТЕРАТУРЫ**

1. Dart // ВикипедиЯ свободная энциклопедия, 2018 : [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Dart.org (дата обращения: 15.05.2018).
2. Dart: как правильно начать знакомство с языком? // Habr, 2017 : [Электронный ресурс]. – Режим доступа: https://habr.com/company/ wrike/blog/ 343988/ (дата обращения: 15.05.2018).
3. Flutter (software) // ВикипедиЯ свободная энциклопедия, 2018 : [Электронный ресурс]. – Режим доступа: https://en.wikipedia.org/wiki/ Flutter\_(software) (дата обращения: 15.05.2017).
4. Пять причин проникнуться симпатией к Flutter // Habr, 2018 : [Электронный ресурс]. – Режим доступа: https://habr.com/company/ ruvds/blog/349622/ (дата обращения: 15.05.2018).
5. Downloads // Git, 2018 : [Электронный ресурс]. – Режим доступа: https://git-scm.com/downloads (дата обращения: 15.05.2018).
6. Why Dart? // Dart, 2018 : [Электронный ресурс]. – Режим доступа: https://git-scm.com/downloads (дата обращения: 15.05.2017).
7. Getting Started // Visual Studio Code, 2018 : [Электронный ресурс]. – Режим доступа: https://code.visualstudio.com/docs (дата обращения: 15.05.2018).
8. Top Dart packages // Dart Packages, 2018 : [Электронный ресурс]. – Режим доступа: https://pub.dartlang.org (дата обращения: 15.05.2018).
9. DateTime // Dart SDK, 2018 : [Электронный ресурс]. – Режим доступа: https://api.dartlang.org/stable/1.24.3/dart-core/DateTime-class.html (дата обращения: 15.05.2018).
10. DateFormat class // Dart Packages, 2018 : [Электронный ресурс]. – Режим доступа: https://www.dartdocs.org/documentation/intl/latest/intl/ DateFormat-class.html (дата обращения: 15.05.2018).
11. Get Started: Install on Windows // Flutter, 2018 : [Электронный ресурс]. – Режим доступа: https://flutter.io/setup-windows/ (дата обращения: 15.05.2018).
12. Flutter SDK Archive // Flutter, 2018 : [Электронный ресурс]. – Режим доступа: https://flutter.io/sdk-archive/ (дата обращения: 15.05.2018).
13. Android Studio // Developers, 2018 : [Электронный ресурс]. – Режим доступа: https://developer.android.com/studio/index.html (дата обращения: 15.05.2018).
14. Download IntelliJ IDEA // Jet Brains, 2018 : [Электронный ресурс]. – Режим доступа: https://www.jetbrains.com/idea/download/ #section=windows (дата обращения: 15.05.2018).
15. Build beautiful native apps in record time // Flutter, 2018 : [Электронный ресурс]. – Режим доступа: https://flutter.io (дата обращения: 15.05.2018).