ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

***«*САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»**

Институт компьютерных наук и технологий

**Отчет о прохождении технологической практики**

Рубша Анастасия Игоревна

*(Ф.И.О. обучающегося)*

3 курс, гр.3530901/80203

09.03.01 «Информатика и вычислительная техника»

*(Направление подготовки (код и наименование)*

**Место прохождения практики**: Высшая школа интеллектуальных систем и суперкомпьютерных технологий (ВШИСиСТ) ИКНТ ФГАОУ ВО «СПбПУ» с использованием электронного обучения и дистанционных образовательных технологий.

**Сроки практики:** с 05 июня по 03 июля 2021 г.

**Руководитель практики:**

Жуков А.В., к.т.н., доцент ВШИСиСТ ИКНТ

*(Ф.И.О., уч.степень, должность)*

**Оценка (зачет):**

Руководитель практики: Жуков А.В.

Обучающийся: Рубша А.И.

Дата:

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

***«*САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО»**

Институт компьютерных наук и технологий

**ИНДИВИДУАЛЬНЫЙ ПЛАН (ЗАДАНИЕ И ГРАФИК) ПРОВЕДЕНИЯ ПРАКТИКИ**

Ф.И.О. обучающегося Рубша Анастасия Игоревна

**Направление подготовки** (код/наименование): 09.03.01, «Информатика и вычислительная техника»

**Профиль** (код/наименование): 09.03.01\_02, «Технологии разработки программного обеспечения»

**Вид практики:** производственная

**Тип практики:** технологическая

**Место прохождения практики**: Высшая школа интеллектуальных систем и суперкомпьютерных технологий (ВШИСиСТ) ИКНТ ФГАОУ ВО «СПбПУ» с

использованием электронного обучения и дистанционных образовательных технологий.

Руководитель практики: Жуков А.В, к.т.н., доцент ВШИСиСТ ИКНТ

*(Ф.И.О., уч.степень, должность)*

**Рабочий график проведения производственной (технологической) практики**

Сроки практики: с **05.06.2021** г. по **03.07.2021** г.

|  |  |  |  |
| --- | --- | --- | --- |
| №  п/п | Этапы (периоды)  практики | Вид работ | Сроки прохождения этапа  (периода) практики |
| 1 | Организационный этап | Установочная лекция (вебинар) для разъяснения целей, задач, содержания и порядка прохождения практики, выдача сопроводительных документов  по практике  Основная тема: написание приложения (английский разговорник/словарик для туристов) для Android на языке Java | 05.06.2021 г. |
| 2 | Основной этап | Изучение среды разработки Android Studio, а также сторонних библиотек.  Сбор информации, обработка, систематизация и анализ фактического и теоретического материала.  Содержание практики: разработка мобильного приложения (английский разговорник/словарик для туристов).  Планируемые результаты прохождения практики: получение навыков создания мобильных программ под Android и полностью готовое мобильное приложение. | 07.06.2021 - 29.06.2021 г. |
| 3 | Заключительный этап | Подготовка отчета | 29.06 - 01.07.2021 г. |
| Защита отчета по практике (сдача зачета) | 02.07 – 03.07.2021 г |

Обучающийся Рубша А.И.

Руководитель практики Жуков А.В.

**Оглавление**

1. [Техническое задание 4](#_bookmark0)
2. [Метод решения 4](#_bookmark1)
3. [Скриншоты работы программы 6](#_bookmark2)
4. [Список используемых источников 8](#_bookmark11)
5. [Приложение 8](#_bookmark12)

**Техническое задание**

**Цель работы:** изучение среды разработки Android Studio, сторонних библиотек, а также сбор информации, обработка, систематизация и анализ фактического и теоретического материала.

На основе полученных знаний необходимо создать мобильное приложение английский разговорник для туристов, которое будет содержать в себе словарик фраз, которые помогут туристу сориентироваться в англоговорящей стране.

**О приложении:**

Разговорник для туристов – настоящий помощник, который позволит человеку не потеряться и освоиться в англоговорящей стране. Приложение содержит 11 разделов, каждый из которых содержит в себе ряд фраз, которые помогут сориентироваться туристу в какой-либо ситуации.

**Метод решения**

На главном меню есть кнопки «Старт», «О приложении» и «Закрыть». При нажатии на кнопку «О приложении» пользователь сможет ознакомиться с краткой информацией о данном мобильном приложении.

При нажатии на кнопку «Старт» пользователю открывается поле с 11-ю разделами (Ресторан\кафе, Обращение\приветствие, Отель, Узнать дорогу, Банк\деньги, Экскурсии, Такси, Покупки\Магазин, Аэропорт, Больница, Автобус\Поезд). Нажав на каждую из иконок, пользователь будет видеть на экране множество англо-русских фраз, которые относятся к выбранной категории.

Например, нажав на иконку «Такси», пользователь узнает, как сказать на английском «Не могли бы Вы меня отвезти в центр города?», «Сколько времени займет дорога?» либо же «Отвезите меня по этому адресу, пожалуйста».

Мобильное приложение содержит 4 класса, отвечающих за корректную работу приложения. (MainActivity, startActivity, LogicActivity, AboutAct)

Теперь же рассмотрим методы, приведенные в классе **MainActivity**, отвечающего за отображение главного окна данной игры и открытие других окон через нажатие на кнопки, располагающиеся на данном экране.

1. onCreate() – данный метод отображает главное окно приложения – т.е main\_menu.xml
2. startApp() – данный метод производит обработку нажатия на кнопку «Старт». При нажатии на нее пользователь переходит к основному рабочему окну.
3. closeApp() – данный метод производит обработку нажатия на кнопку «Закрыть». При нажатии на нее пользователь полностью закрывает приложение.
4. aboutInfo() - данный метод производит обработку нажатия на кнопку «О приложении». При нажатии на нее пользователь переходит к окну с краткой информацией о приложении.

Рассмотрим методы, приведенные в классе **AboutAct**, отвечающем за отображение окна с правилами игры для пользователя. Класс построен на about\_activity.xml файле.

1. onCreate() – единственный и главный метод, отображающий пользователю окно с правилами игры, основанный на about\_activity.xml

Рассмотрим методы, приведенные в классе **startActivity**, отвечающем за определение в данном окне 12 кнопок, а также их обработку.

1. onCreate() – метод, отображающий окошко, содержащее в себе 12 кнопок (среди которых 11 – разделы, а 12-я – кнопка перехода назад, на прошлое окно). Данное окно основано на main\_cells.xml.
2. back() – обработчик нажатия кнопки «Назад». Нажав на нее, пользователь снова оказывается в главном меню.
3. findViewsById() - инициализирует названия кнопок, через поиск по их id в main\_cells.xml файле
4. onClick() – производит обработку нажатия всех 11-ти кнопок. При нажатии на определенную иконку, переменная indOfCell присваивает себе определенное значение (в зависимости от того, какую категорию выбрал пользователь). Данная переменная нам нужна для дальнейшей работы, она будет использоваться в классе, который мы рассмотрим далее.

Теперь рассмотрим методы, приведенные в классе **LogicActivity**, отвечающем за чтение наших файлов и формирования из него списка фраз после того, как пользователь выбрал определенную категорию.

1. onCreate() – данный метод в данном случае не отображает новое окно, однако он содержит в себе следующий метод – readText().
2. readText() – метод, работающий с переменной (indOfCell), определенной из прошлого класса – startActivity. В зависимости от значения данной переменной, наша программа определяет, с каким .txt файлом он будет работать.
3. makeList() – метод, который производит чтение .txt файла и приведения его к виду списка, который пользователь может листать. Весомую помощь в приведении к виду списка оказывает класс ArrayAdapter.

Класс **ArrayAdapter** — это простейший адаптер, связывающий массив данных с набором компонентов TextView, из которых, например, и может состоять ListView. Таким образом, в нашем случае источник данных — массив объектов. При этом ArrayAdapter вызывает метод toString() у каждого объекта (что необходимо для приведения к строковому виду), а полученную строку устанавливает в TextView.

Теперь же покажем скриншоты полученного нами приложения.

**Скриншоты работы приложения.**

****

Рис. 1-2 Отображение окна главного меню и окна информации о приложении.

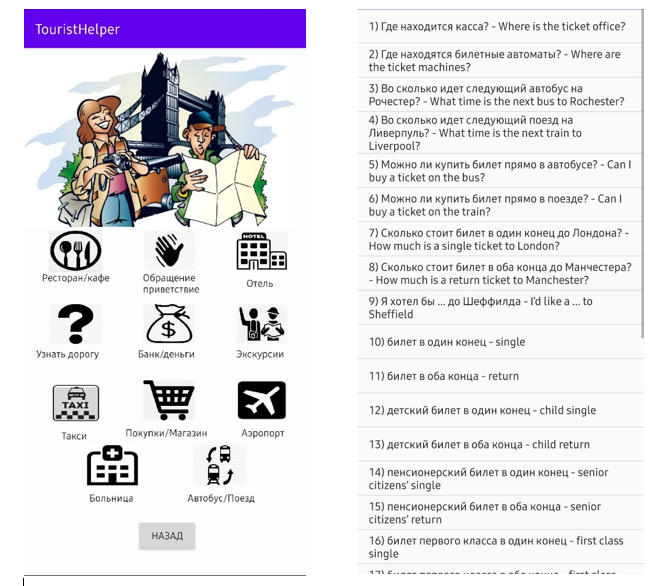


Рис. 3-4 Отображение окна с выбором категорий и содержимого категории «Автобус\Поезд»

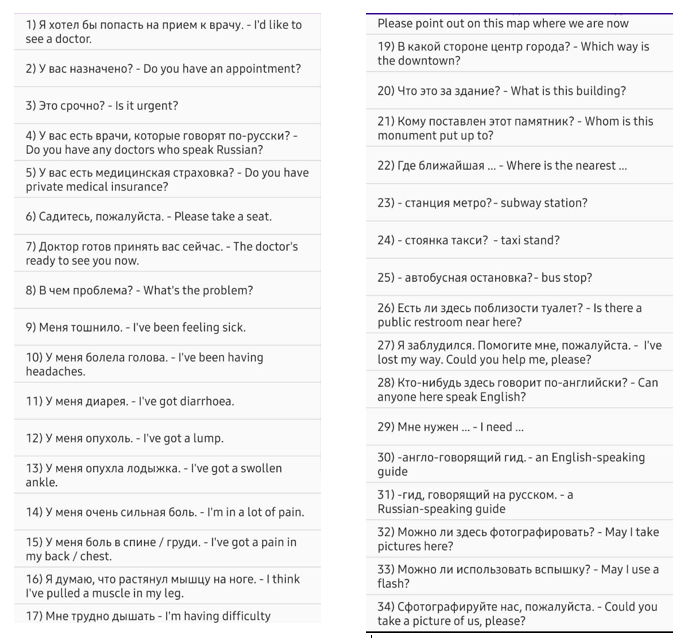


Рис. 5-6 Отображение содержимого категории «Больница» и содержимого категории «Экскурсии»

**Список использованных источников.**

1. https://otus.ru/nest/post/1955/
2. http://developer.alexanderklimov.ru/android/theory/arrayadapter.php
3. https://metanit.com/java/android/5.1.php

**Приложение**

1. **MainActivity**

public class MainActivity extends AppCompatActivity {  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*main\_menu*); //метод отображения главного окна  
 setRequestedOrientation(ActivityInfo.*SCREEN\_ORIENTATION\_PORTRAIT*); //сохранение вертикальной ориентации окна  
 }  
 public void startApp(View v){ //метод-обработчик кнопки "Старт"  
 Intent startAct = new Intent(MainActivity.this, startActivity.class); //при нажатии переход к классу startActivity  
 startActivity(startAct);  
 }  
 public void aboutInfo(View v){ //метод-обработчик кнопки "О приложении"  
 Intent aboutInfo = new Intent(MainActivity.this, AboutAct.class); //при нажатии переход к классу AboutAct  
 startActivity(aboutInfo);  
 }  
 public void closeApp(View v){ //метод-обработчик кнопки "Закрыть"  
 moveTaskToBack(true); //по нажатию на кнопку приложение закрывается  
 finish();  
 System.*exit*(0);  
 }  
}

1. **AboutAct**

public class AboutAct extends AppCompatActivity {  
 @Override  
 protected void onCreate(Bundle savedInstanceState) { //открытие окна с правилами игры, при нажатии на кнопку About.  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*about\_activity*); //отображение окна с информацией о приложении setRequestedOrientation(ActivityInfo.*SCREEN\_ORIENTATION\_PORTRAIT*);//сохранение вертикальной ориентации окна  
 }  
}

1. **startActivity**

public class startActivity extends AppCompatActivity implements View.OnClickListener {  
 @Override  
 protected void onCreate(Bundle savedInstanceState) { //открытие окна с правилами игры, при нажатии на кнопку About.  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*main\_cells*);//отображение окна с выбором категорий  
 setRequestedOrientation(ActivityInfo.*SCREEN\_ORIENTATION\_PORTRAIT*);//сохранение вертикальной ориентации окна  
 findViewsById();  
 }  
 public void back(View v){ //метод-обработчик кнопки "Назад" в окне с выбором категорий  
 Intent back = new Intent(startActivity.this, MainActivity.class);  
 startActivity(back);//при нажатии "назад" пользователь перейдет в главное меню  
 }  
 public void findViewsById(){ //метод который определяет каждой кнопке его id в рабочем окне  
 View buttonFood = findViewById(R.id.*food*);  
 buttonFood.setOnClickListener(this); //обработчик нажатия на кнопки  
 View buttonMeeting = findViewById(R.id.*findRoad*);  
 buttonMeeting.setOnClickListener(this);  
 View buttonTour = findViewById(R.id.*tour*);  
 buttonTour.setOnClickListener(this);  
 View buttonHotel = findViewById(R.id.*hotel*);  
 buttonHotel.setOnClickListener(this);  
 View buttonHello = findViewById(R.id.*hello*);  
 buttonHello.setOnClickListener(this);  
 View buttonMoney = findViewById(R.id.*money*);  
 buttonMoney.setOnClickListener(this);  
 View buttonAirport = findViewById(R.id.*airport*);  
 buttonAirport.setOnClickListener(this);  
 View buttonHospital = findViewById(R.id.*hospital*);  
 buttonHospital.setOnClickListener(this);  
 View buttonTaxi = findViewById(R.id.*taxi*);  
 buttonTaxi.setOnClickListener(this);  
 View buttonShopping = findViewById(R.id.*shopping*);  
 buttonShopping.setOnClickListener(this);  
 View buttonBusTrain = findViewById(R.id.*busTrain*);  
 buttonBusTrain.setOnClickListener(this);  
 }  
 public static int *indOfCell*; //переменная класса, которой мы присваиваем значение в зависимости от выбранной категории  
 public void onClick(View view) { //метод-обработчик нажатия кнопок  
 if (view.getId() == (R.id.*food*)) { *indOfCell* = 1; } //присваивание переменной значения при нажатии  
 if (view.getId() == (R.id.*findRoad*)) {*indOfCell* = 4;}  
 if (view.getId() == (R.id.*tour*)) { *indOfCell* = 6; }  
 if (view.getId() == (R.id.*hotel*)) { *indOfCell* = 3; }  
 if (view.getId() == (R.id.*hello*)) { *indOfCell* = 2; }  
 if (view.getId() == (R.id.*money*)) { *indOfCell* = 5; }  
 if (view.getId() == (R.id.*taxi*)) { *indOfCell* = 7; }  
 if (view.getId() == (R.id.*shopping*)) { *indOfCell* = 8; }  
 if (view.getId() == (R.id.*airport*)) { *indOfCell* = 9; }  
 if (view.getId() == (R.id.*hospital*)) { *indOfCell* = 10; }  
 if (view.getId() == (R.id.*busTrain*)) { *indOfCell* = 11; }  
 Intent logicActivity = new Intent(this, LogicActivity.class);   
 startActivity(logicActivity); //осуществляется переход в класс LogicActivity  
 }   
}

1. **LogicActivity**

public class LogicActivity extends ListActivity {  
 ArrayAdapter<String> mAdapter = null;//создаем адаптер для дальнейшей работы с выстраиванием списков  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {//метод запускающий работу читки тхт файла  
 super.onCreate(savedInstanceState);  
 try {  
 readText();//вызов метода readText  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 public void readText() throws IOException {//метод, который открывает определенный тхт файл в зависимости от значения переменной  
 if (*indOfCell* == 1) { //которую мы получили в классе startActivity  
 String[] food = new String[10]; //создание пустого массива определенного размера для дальнейшей работы в нем  
 makeList("food.txt", food);//вызов метода makeList с переменными "food.txt" и food  
 }  
 if (*indOfCell* == 2) {  
 String[] hello = new String[30];  
 makeList("hello.txt", hello);  
 }  
 if (*indOfCell* == 3) {  
 String[] hotel = new String[12];  
 makeList("hotel.txt", hotel);  
 }  
 if (*indOfCell* == 4) {  
 String[] findRoad = new String[9] ;  
 makeList("findRoad.txt", findRoad);  
 }  
 if (*indOfCell* == 5) {  
 String[] money = new String[12] ;  
 makeList("money.txt", money);  
 }  
 if (*indOfCell* == 6){  
 String[] tour = new String[34];  
 makeList("tour.txt", tour);  
 }  
 if (*indOfCell* == 7) {  
 String[] taxi = new String[15];  
 makeList("taxi.txt", taxi);  
 }  
 if (*indOfCell* == 8) {  
 String[] shopping = new String[15];  
 makeList("shopping.txt", shopping);  
 }  
 if (*indOfCell* == 9) {  
 String[] airport = new String[18] ;  
 makeList("airport.txt", airport);  
 }  
 if (*indOfCell* == 10) {  
 String[] hospital = new String[22] ;  
 makeList("hospital.txt", hospital);  
 }  
 if (*indOfCell* == 11){  
 String[] busTrain = new String[28];  
 makeList("busTrain.txt", busTrain);  
 }  
 }  
 public void makeList(String filename, String[] cell) throws IOException {//метод, преобразующий тхт файл в прокручивающийся список строк  
 InputStream is = getAssets().open(filename);//определение тхт файл из папки assets  
 int size = is.available(); // возвращение количество байтов ввода, доступные в данный момент для чтения  
 byte[] buffer = new byte[size];//создание массива байтов размера size, доступные в данный момент для чтения  
 is.read(buffer);//чтение байт в буфер, возвращая количество прочитанных байтов. По достижении конца файла возвращает значение -1  
 String arr = new String(buffer);//перевод массива байтов в строку для чтения  
 for (int i = 0; i < Arrays.*asList*(arr.split("\n")).size(); i++) {//заполнение пустого массива cell значениями из тхт файла  
 cell[i] = Arrays.*asList*(arr.split("\n")).get(i);  
 }  
 mAdapter = new ArrayAdapter<>(this, android.R.layout.*simple\_list\_item\_1*, cell);//преобразование массива в прокручиваемый список  
  
 is.close();//закрытие источник ввода. Следующие попытки чтения передадут исключение IOException  
 setListAdapter(mAdapter);//подключение адаптера  
 }  
}