Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра вычислительных методов и программирования

Отчёт по лабораторной работе № 2

по дисциплине «Программируемые мобильные устройства»

**Интенты. Интент-фильтры**

Вариант № 10

Проверил: Выполнил:

Потапчик В. С. Студент гр.724402

Игнатьева В.Ю.

Минск, 2020

1. **Цель лабораторной работы**

Приобрести навыки работы с Activity. Создать многооконное приложение и передавать данные между окнами.

1. **Теоретические сведения**

2.1 Объекты Intent

[Intent](https://developer.android.com/reference/android/content/Intent?hl=ru) представляет собой объект обмена сообщениями, с помощью которого можно запросить выполнение действия у [компонента другого приложения](https://developer.android.com/guide/components/fundamentals?hl=ru#Components). Несмотря на то, что объекты Intent упрощают обмен данными между компонентами по нескольким аспектам, в основном они используются в трех ситуациях:

* Для запуска операции

Компонент [Activity](https://developer.android.com/reference/android/app/Activity?hl=ru) представляет собой один экран в приложении. Для запуска нового экземпляра компонента [Activity](https://developer.android.com/reference/android/app/Activity?hl=ru) необходимо передать объект [Intent](https://developer.android.com/reference/android/content/Intent?hl=ru) методу [startActivity()](https://developer.android.com/reference/android/content/Context?hl=ru#startActivity(android.content.Intent)). Объект [Intent](https://developer.android.com/reference/android/content/Intent?hl=ru) описывает операцию, которую требуется запустить, а также содержит все остальные необходимые данные.

* Для запуска службы

[Service](https://developer.android.com/reference/android/app/Service?hl=ru) является компонентом, который выполняет действия в фоновом режиме без пользовательского интерфейса. Службу можно запустить для выполнения однократного действия (например, чтобы загрузить файл), передав объект[Intent](https://developer.android.com/reference/android/content/Intent?hl=ru) методу [startService()](https://developer.android.com/reference/android/content/Context?hl=ru#startService(android.content.Intent)). Объект [Intent](https://developer.android.com/reference/android/content/Intent?hl=ru) описывает службу, которую требуется запустить, а также содержит все остальные необходимые данные.

* Для рассылки широковещательных сообщений:

Широковещательное сообщение – это сообщение, которое может принять любое приложение. Система выдает различные широковещательные сообщения о системных событиях, например, когда система загружается или устройство начинает заряжаться. Для выдачи широковещательных сообщений другим приложениям необходимо передать объект [Intent](https://developer.android.com/reference/android/content/Intent?hl=ru) методу [sendBroadcast()](https://developer.android.com/reference/android/content/Context?hl=ru#sendBroadcast(android.content.Intent)), [sendOrderedBroadcast()](https://developer.android.com/reference/android/content/Context?hl=ru#sendOrderedBroadcast(android.content.Intent,%20java.lang.String)) или [sendStickyBroadcast()](https://developer.android.com/reference/android/content/Context?hl=ru#sendStickyBroadcast(android.content.Intent)).

2.2 Типы объектов Intent

Есть два типа объектов Intent:

1. **Явные объекты Intent** указывают компонент, который требуется запустить, по имени (полное имя класса). Явные объекты Intent обычно используются для запуска компонента из этого же приложения, поскольку известно имя класса операции или службы, которую необходимо запустить.
2. **Неявные объекты Intent** не содержат имени конкретного компонента. Вместо этого они в целом объявляют действие, которое требуется выполнить, что дает возможность компоненту из другого приложения обработать этот запрос. Например, если требуется показать пользователю место на карте, то с помощью неявного объекта Intent можно запросить, чтобы это сделало другое приложение, в котором такая возможность предусмотрена.
   1. Интент-фильтры

Фильтр Intent представляет собой выражение в файле манифеста приложения, указывающее типы объектов Intent, которые мог бы принимать компонент. Например, объявив фильтр Intent для операции, вы даете другим приложениям возможность напрямую запускать вашу операцию с помощью некоторого объекта Intent. Точно так же, если *не* объявлять какие-либо фильтры Intent для операции, то ее можно будет запустить только с помощью явного объекта Intent.

В фильтре намерений декларируется только три составляющих объекта Intent: действие, данные, категория. Дополнения и флаги не играют никакой роли в принятии решения, какой компонент получает намерение.

Внутри узла Фильтра намерений можно использовать такие теги (и соответствующие атрибуты).

* Action – Использует атрибут android:name, чтобы задать название действия, которое будет обслуживаться. Каждый фильтр намерений должен иметь один (и только один) тег action. Действия должны быть представлены в виде уникальных строк, которые сами себя описывают.
* Сategory – Использует атрибут android:name, чтобы указать, при каких обстоятельствах должно обслуживаться действие. Каждый тег intent-filter способен содержать несколько тегов category.
* Data – Этот тег дает возможность указать тип данных, с которым может взаимодействовать компонент. При необходимости можно задать несколько тегов data.
  1. Принцип работы фильтров намерений

При использовании метода startActivity() передаваемое неявное намерение, как правило, доходит лишь до одной активности. Если для выполнения заданного действия с указанными данными годятся сразу несколько активностей, пользователю предоставляется список выбора. Процесс, когда решается, какую активность лучше запустить, называется Утверждением намерений. Его цель – найти наиболее подходящий фильтр намерений. В целом весь алгоритм работает следующим образом.

1. Android собирает список всех доступных Фильтров намерений из установленных пакетов.
2. Фильтры, которые не соответствуют действию или категории Намерения, удаляются из списка. Совпадение происходит только в том случае, если Фильтр намерений содержит указанное действие (или если действие для него вовсе не задано). Совпадения не произойдёт, только если ни одно из действий Фильтра намерений не будет эквивалентно тому, которое задано в Намерении. Для категорий процесс соответствия более строгий. Фильтр намерений должен включать в себя все категории, заданные в полученном Намерении. Фильтр, для которого категории не указаны, может соответствовать только таким же Намерениям (нет категорий).
3. Наконец, каждая часть пути URI из Намерения сравнивается с тегом data Фильтра намерений. Если в Фильтре указаны схема (протокол), сервер/принадлежность, путь или тип MIME, все эти значения проверяются на соответствие пути URI из Намерения. При любом несовпадении Фильтр будет удален из списка. Если в Фильтре намерений не указано ни одного параметра data, его действие будет распространяться на любые данные.
4. Когда неявным образом запускается Активность и вышеописанный процесс возвращает более одного совпадения, пользователю выводится список со всеми вариантами.
5. **Пример работы приложения**

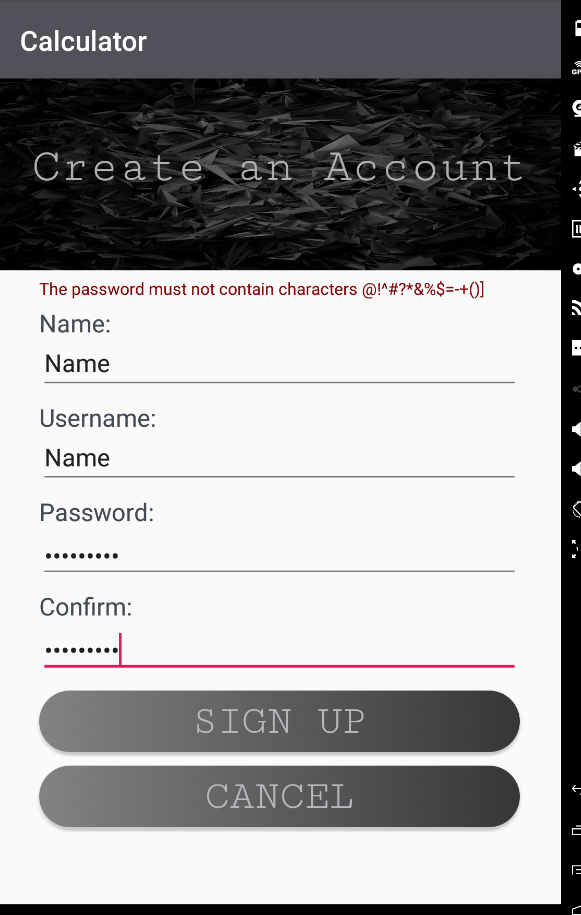
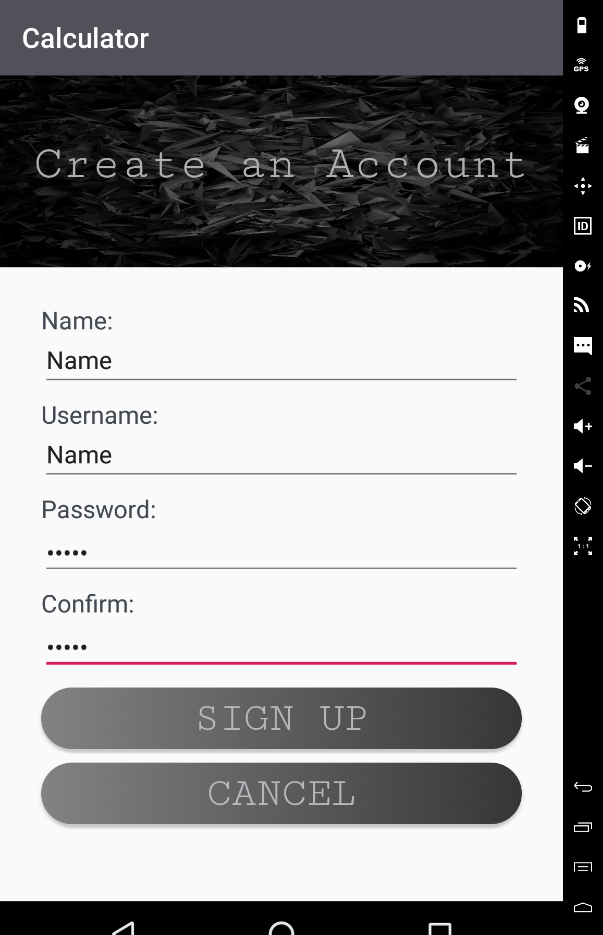


Рисунок 3.1 – Окно регистрации

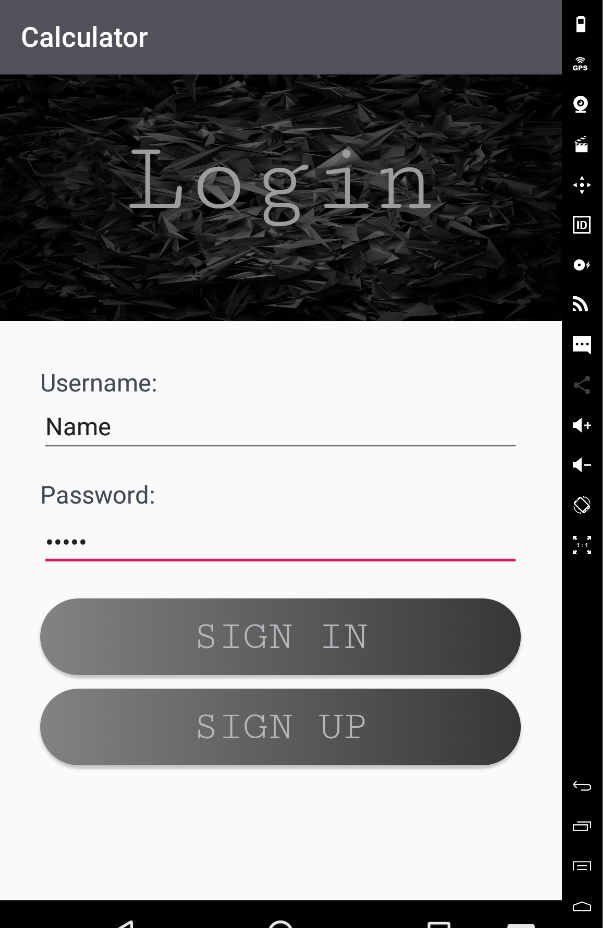
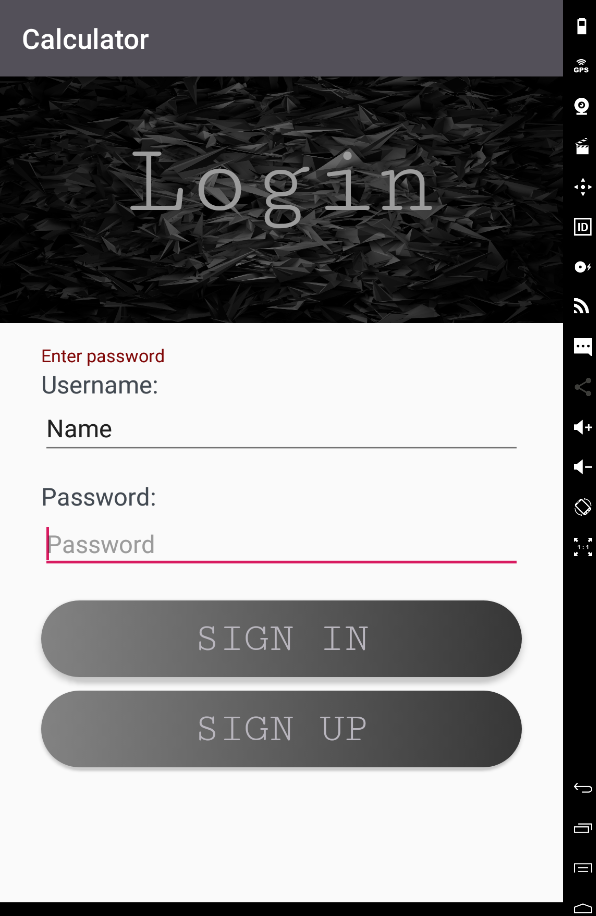
 

Рисунок 3.2 – Окно входа

1. **Листинг кода программы**

Класс RegisterActivity:

public class RegisterActivity extends AppCompatActivity implements View.OnClickListener, View.OnTouchListener, View.OnKeyListener{  
  
 EditText textName, textUsername, textPassword, textPasswordConfirm;  
 Button buttonSignUp, buttonCancel;  
 TextView textError;  
  
 public static final String APP\_PREFERENCES = "accounts";  
 public static final String APP\_PREFERENCES\_NAME = "Name";  
 public static final String APP\_PREFERENCES\_USERNAME = "Username";  
 public static final String APP\_PREFERENCES\_PASSWORD = "Password";  
 SharedPreferences account;  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_register);  
  
 account = getSharedPreferences(APP\_PREFERENCES, Context.MODE\_PRIVATE);   
 buttonSignUp = (Button) findViewById(R.id.signUp);  
 buttonSignUp.setOnClickListener(this);  
 buttonCancel = (Button) findViewById(R.id.signUpCancel);  
 buttonCancel.setOnClickListener(this);  
 textUsername = (EditText) findViewById(R.id.username);  
 textPassword = (EditText) findViewById(R.id.password);  
 textName = (EditText) findViewById(R.id.name);  
 textPasswordConfirm = (EditText) findViewById(R.id.passwordConfirm);  
 textError = (TextView) findViewById(R.id.textErrorReg);  
  
 View.OnKeyListener enterListener = new View.OnKeyListener() {  
 @Override  
 public boolean onKey(View v, int keyCode, KeyEvent event) {  
 if(event.getAction() == KeyEvent.ACTION\_DOWN &&event.getKeyCode() == KeyEvent.KEYCODE\_ENTER){  
 if(textName.hasFocus()){  
 textName.clearFocus();  
 textUsername.requestFocus();  
 return true;  
 }  
 if(textUsername.hasFocus()){  
 textUsername.clearFocus();  
 textPassword.requestFocus();  
 return true;  
 }  
 }  
 return false;  
 }  
 };  
  
 textName.setOnKeyListener(enterListener);  
 textUsername.setOnKeyListener(enterListener);  
 textPassword.setOnKeyListener(enterListener);  
  
 View.OnTouchListener errorListener = new View.OnTouchListener() {  
 @Override  
 public boolean onTouch(View v, MotionEvent event) {  
 textError.setText("");  
 return false;  
 }  
 };  
 textError.setOnTouchListener(errorListener);  
 }  
  
 @Override  
 public void onClick (View v) {  
 switch (v.getId()) {  
 case R.id.signUp:  
 EditText editName = findViewById(R.id.name);  
 String strName = editName.getText().toString();  
 EditText editUserName = findViewById(R.id.username);  
 String strUserName = editUserName.getText().toString();  
 EditText editPassword = findViewById(R.id.password);  
 String strPassword = editPassword.getText().toString();  
 EditText editPasswordConfirm = findViewById(R.id.passwordConfirm);  
 String strPasswordConfirm = editPasswordConfirm.getText().toString();  
  
 if(strName.equals("") || strUserName.equals("") || strPassword.equals("") || strPasswordConfirm.equals("")){  
 textError.setText("Fill in all the fields.");  
 } else{  
 String valid = isValidPassword(strPassword);  
 if(!valid.equals("")){  
 textError.setText(valid);  
 } else{  
 if(strPassword.equals(strPasswordConfirm)) {  
 if(strUserName.equals(account.getString(APP\_PREFERENCES\_USERNAME, ""))){  
 textError.setText("This username is already taken.");  
 } else{  
 SharedPreferences.Editor editor = account.edit();  
 editor.putString(APP\_PREFERENCES\_NAME, strName);  
 editor.putString(APP\_PREFERENCES\_USERNAME, strUserName);  
 editor.putString(APP\_PREFERENCES\_PASSWORD, strPassword);  
 editor.apply();  
 finish();  
 }  
 } else{  
 textError.setText("Passwords do not match.");  
 }  
 }  
 }  
 break;  
 case R.id.signUpCancel:  
 finish();  
 break;  
 default:  
 break;  
 }  
 }  
  
 private boolean isValidPasswordNumbers(String password){  
 Pattern pattern;  
 Matcher matcher;  
 String PASSWORD\_PATTERN\_NUMBERS = "[0-9]";  
 pattern = Pattern.compile(PASSWORD\_PATTERN\_NUMBERS);  
 matcher = pattern.matcher(password);  
 return matcher.find();  
 }  
 private boolean isValidPasswordSymbols(String password){  
 Pattern pattern;  
 Matcher matcher;  
 String PASSWORD\_PATTERN\_SYMBOLS = "[@!^#?\*&%$=+()]";  
 pattern = Pattern.compile(PASSWORD\_PATTERN\_SYMBOLS);  
 matcher = pattern.matcher(password);  
 return matcher.find();  
 }  
 private boolean isValidPasswordLettersSmall(String password){  
 Pattern pattern;  
 Matcher matcher;  
 String PASSWORD\_PATTERN\_LETTERS\_SMALL = "[a-z]";  
 pattern = Pattern.compile(PASSWORD\_PATTERN\_LETTERS\_SMALL);  
 matcher = pattern.matcher(password);  
 return matcher.find();  
 }  
 public String isValidPassword(String password) {  
 String result = "";  
 if(password.length() < 4 || password.length() > 10){  
 result = "Invalid password length";  
 }  
 if(!isValidPasswordNumbers(password)) {  
 result = "The password must contain numbers";  
 }  
 if(!isValidPasswordLetters(password)) {  
 result = "The password must contain capital letters";  
 }  
 if(!isValidPasswordLettersSmall(password)) {  
 result = "The password must contain small letters";  
 }  
 if(isValidPasswordSymbols(password)) {  
 result = "The password must not contain characters @!^#?\*&%$=-+()]";  
 }  
 return result;  
 }  
  
 @Override  
 public boolean onKey(View v, int keyCode, KeyEvent event) {  
 return false;  
 }  
  
 @Override  
 public boolean onTouch(View v, MotionEvent event) {  
 return false;  
 }  
}

Ссылка на репозиторий: <https://github.com/VladaIgnatyeva/labs-Android/tree/master/lab2>