Лабораторна робота №6. Обробка натиснення кнопки та робота з віджетами в java-класах

Мета: вивчити порядок обробки натиснення кнопки та керування елементами View в java-класах.

Теоретичні відомості

При обробці натиснення кнопки чи іншого об’єкта View можна застосувати декілька підходів.

- Визначення атрибута **android:onClick**. Для цього в xml відповідного layout в секції <Button> потрібно визначити атрибут **android:onClick**. Значення цього атрибута повинно бути ім’я метода, який буде оброблювати натиснення (рис. 1).

```
<?xml version="1.0" encoding="utf-8"?>
<Button xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/button_send"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/button_send"
    android:onClick="sendMessage" />
```

Рис. 1

Відповідно java-клас макет повинен містити реалізацію метода обробки натиснення (рис. 2).

```
/** Called when the user touches the button */
public void sendMessage(View view) {
    // Do something in response to button click
}
```

Рис. 2

- Використання **OnClickListener**. Для використання обробника події (event handler) створюєьтся об’єкт **View.OnClickListener**, який асоціюється з віджетом за допомогою метода **setOnClickListener(View.OnClickListener)** (рис. 3).

```
Button button = (Button) findViewById(R.id.button_send);
button.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v) {
        // Do something in response to button click
    }
});
```

Рис. 3

Наприклад, обробка натиснення трьох кнопок та вивід у TextView імені натиснутої кнопки може бути виконана наступним чином

```
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class MainActivity extends AppCompatActivity {

    Button button1;
    Button button2;
    Button button3;
    TextView myTextView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        myTextView = (TextView) findViewById(R.id.textView);
        button1 = (Button) findViewById(R.id.button);
        button2 = (Button) findViewById(R.id.button2);
        button3 = (Button) findViewById(R.id.button3);

        View.OnClickListener onClickListener = new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                switch (v.getId()){
                    case R.id.button: myTextView.setText(R.string.str_btn1);
break;
                    case R.id.button2: myTextView.setText(R.string.str_btn2);
break;
                    case R.id.button3: myTextView.setText(R.string.str_btn3);
break;
                    case R.id.textView: myTextView.setText(R.string.str_txtvwr);
break;

                }
            }
        };

        button1.setOnClickListener(onClickListener);
        button2.setOnClickListener(onClickListener);
        button3.setOnClickListener(onClickListener);
        myTextView.setOnClickListener(onClickListener);
```

```
    }
}
```

Аналогічним чином оброблюється натиснення інших об’єктів View, для яких ввімкнена властивість **android:clickable**.

Підчас роботи додатку у java-класах можливо змінювати атрибути елементів UI та викликати їх специфічні методи.

Наприклад, доступ до деяких властивостей віджетів TextView Button CheckBox продемонстровано нижче.

```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    TextView myTextVeiw= (TextView) findViewById(R.id.myText);
    myTextVeiw.setText("Lab 7.Test");
    Button myButton=(Button) findViewById(R.id.myButton);
    myButton.setText("My new button");
    CheckBox mycheckBox = (CheckBox) findViewById(R.id.mycheckBox);
    mycheckBox.setChecked(true);
}
```

Розглянемо далі деякі специфічні дії, пов’язані з різними елементами View.

Визначення типу екранної клавіатури для TextView. Android пропонує декілька типів клавіатур для вводу в текстові поля. Для специфікації типу клавіатури використовується атрибут **android:inputType** у xml файлі (рис. 1).

```
<EditText
    android:id="@+id/email_address"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="@string/email_hint"
    android:inputType="textEmailAddress" />
```

Рис. 1

Можливі види клавіатури та відповідні значення **android:inputType** наведено нижче.

Рис. 2. android:inputType=”text”

Рис. 3. android:inputType=”textEmailAddress”

Рис. 4. android:inputType=”phone”

Можливі значення атрибута **android:inputType**:

- "**text**" – звичайна клавіатура (рис. 2);
- "**textEmailAddress**" – звичайна клавіатура з символом @;
- "**textUri**" – звичайна клавіатура з символом /;
- "**number**" – цифрова клавіатура;
- "**phone**" – телефонна клавіатура.

Визначення деякої специфічної поведінки екранної клавіатури. В атрибуті **android:inputType** також можна задавати певні види поведінки клавіатури. Деякі можливі параметри:
"**textCapSentences**" – звичайна клавіатура, яка кожне нове речення пише з великої літери;
"**textCapWords**" – звичайна клавіатура, яка кожне слово пише з великої літери;
"**textAutoCorrect**" – звичайна клавіатура з виправленням слів з помилками;
"**textPassword**" – звичайна клавіатура зі скритим зірочками текстом;
"**textMultiLine**" – звичайна клавіатура, яка може включати великі строки з розривом строки.

```
<EditText
    android:id="@+id/postal_address"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="@string/postal_address_hint"
    android:inputType="textPostalAddress|
                       textCapWords|
                       textNoSuggestions" />
```

Рис. 5

Специфікація кнопки дії на екранній клавіатурі. Після завершення вводу зазвичай потрібно виконати якусь дію, наприклад, виконати пересилку “Send” або пошук “Search”. Для цього потрібно визначити атрибут **android:imeOptions** (рис. 6).

```
<EditText
    android:id="@+id/search"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:hint="@string/search_hint"
    android:inputType="text"
    android:imeOptions="actionSend" />
```

Рис. 6

Обробка натиснення кнопки дії на клавіатурі. Для обробки натиснення на кнопку дії на клавіатурі застосовується інтерфейс **TextView.OnEditorActionListener** з визначеним методом **onEditorAction()**, який ідентифікує який тип дії виконано (рис. 7)

```
EditText editText = (EditText) findViewById(R.id.search);
editText.setOnEditorActionListener(new OnEditorActionListener() {
    @Override
    public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {
        boolean handled = false;
        if (actionId == EditorInfo.IME_ACTION_SEND) {
            sendMessage();
            handled = true;
        }
        return handled;
    }
});
```

Рис. 7

Завдання до лабораторної роботи

1. У розробленому в лабораторній роботі №5 (пункт 11) додатку специфікуйте різні види клавіатури для вводу імені, електронної пошти, пароля і т.д.
2. Додайте оброблювач натиснення кнопки «Can’t wait. Sign me up!» з використанням обробника подій **OnClickListener**. Після натиснення запускається друга activity з TextView, яка містить введені користувачем строки.

Контрольні запитання

1. Які основні способи обробки натиснення кнопки?
2. Які специфічні атрибути та методи визначені для елемента TextView?
3. Що таке кнопка дії на екранній клавіатурі та яка обробити її натиснення?