Лабораторна робота №6. Робота з меню

Мета: вивчити порядок створення та роботи з меню в Android.

Теоретичні відомості

Android додатках використовується декілька типів меню, які базуються на API-інтерфейсах класу **Menu**.

Зазвичай використовуються три типи меню:

- Меню параметрів.
- Контекстне меню.
- Спливаюче меню.

Для визначення пунктів всіх типів меню в Android редагується xml файл в папці **res/menu/**. Після цього ресурс з пунктами меню можна завантажити як об’єкт Menu до відповідного activity.

Для визначення меню в xml використовуються наступні елементи.

<menu>. Визначає клас **Menu**, який є контейнером для пунктів меню. Елемент **<menu>** повинен бути кореневим елементом в xml файлі, в ньому має бути один або декілька елементів **<item>** та **<group>**.

<item>. Створює клас **MenuItem**, який визначає один пункт меню. Цей елемент може містити вкладені елементи **<menu>** для створення вкладених розділів меню.

<group>. Необов’язковий невидимий контейнер для елементів **<item>**, який може містити інші елементи, дозволяє розбивати пункти меню на категорії та призначати їм однакові властивості.

Структура xml файла ресурсу меню може мати наступний вигляд:

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@[+][package:]id/resource_name"
          android:title="string"
          android:titleCondensed="string"
          android:icon="@[package:]drawable/drawable_resource_name"
          android:onClick="method name"
          android:showAsAction=["ifRoom" | "never" | "withText" | "always" |
"collapseActionView"]
          android:actionLayout="@[package:]layout/layout_resource_name"
          android:actionViewClass="class name"
```

```
            android:actionProviderClass="class name"
            android:alphabeticShortcut="string"
            android:numericShortcut="string"
            android:checkable=["true" | "false"]
            android:visible=["true" | "false"]
            android:enabled=["true" | "false"]
            android:menuCategory=["container" | "system" | "secondary" | "alternative"]
            android:orderInCategory="integer" />
    <group android:id="@[+][package:]id/resource name"
           android:checkableBehavior=["none" | "all" | "single"]
           android:visible=["true" | "false"]
           android:enabled=["true" | "false"]
           android:menuCategory=["container" | "system" | "secondary" | "alternative"]
           android:orderInCategory="integer" >
        <item />
    </group>
    <item >
        <menu>
          <item />
        </menu>
    </item>
</menu>
```

Атрибути елемента <item>.

android:title. Строковий ресурс. Визначає заголовок меню.

android:titleCondensed. Строковий ресурс. Заголовок меню, який використовується коли повна назва не може відобразитись повністю.

android:icon. Drawable ресурс. Зображення іконки меню.

android:onClick. Ім'я методу, який буде викликатися при натисненні на пункт меню. Метод повинен бути визначеним в класі activity як public та приймати єдиний параметр **MenuItem**, який ідентифікує викликаний пункт меню.

android:showAsAction. Атрибут, який вказує на те, коли і як повинен відображатися певний пункт меню в **рядку дій (app bar або action bar).**

Значення	**Опис**
ifRoom	Відображати об'єкт тільки тоді, коли для цього є місце в рядку дій. Якщо немає місця для всіх пунктів з атрибутом «ifRoom» відображається пункт меню з нижчим атрибутом **orderInCategory**, а ішні пункти відображаються в спливаючому меню.
withText	Включає текст заголовка (атрибут

	android:title)
Never	Ніколи не відображати меню на **app bar**. Пункти меню відображаються в спливаючому меню.
Always	Завжди відображати елемент на **app bar**.
collapseActionView	Віджет, який пов'язано з даним пунктом меню може згортатися.

android:actionLayout. Ресурс макета. Визначає layout, який пов'язано з віджетом.

android:alphabeticShortcut. Символ для виклику меню по гарячій клавіші.

android:numericShortcut. Число для виклику меню по числовій гарячій клавіші.

Наприклад, файл меню з назвою game_menu може мати наступний вигляд (game_menu.xml):

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/new_game"
          android:icon="@drawable/ic_new_game"
          android:title="@string/new_game"
          android:showAsAction="ifRoom"/>
    <item android:id="@+id/help"
          android:icon="@drawable/ic_help"
          android:title="@string/help" />
</menu>
```

Рис. 1

До будь-якого пункта меню (крім вкладеного) можна додати вкладене меню, додавши елемент **<menu>** як підлеглий до пункту **<item>** (рис. 2).

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:id="@+id/file"
          android:title="@string/file" >
        <!-- "file" submenu -->
        <menu>
            <item android:id="@+id/create_new"
                  android:title="@string/create_new" />
            <item android:id="@+id/open"
                  android:title="@string/open" />
        </menu>
    </item>
</menu>
```

Рис. 2

Створення меню параметрів. В меню параметрів розташовуються найбільш важливі для поточного контексту дії.

Місце, де відображається меню параметрів залежить від версії Android API. У версії Android 3.0 (рівень API 11) і вище, пункти меню параметрів відображаються у рядку дій (**action bar**) (рис. 3).

Рис. 3

За замовчуванням, система розташовує всі пункти меню на панелі додаткових варіантів, яка розташовується праворуч. Для швидкого доступу до деяких дій можливо примусово розташувати їх на рядку дій за допомогою значення **android:showAsAction="ifRoom"**.

Для того, щоб вказати пункти меню у Activity, потрібно перевизначити метод **onCreateOptionsMenu()**. В цьому методі потрібно завантажити створений ресурс меню (визначений в xml) в клас **Menu** (рис. 4).

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.game_menu, menu);
    return true;
}
```

Рис. 4

Обробка натискань. Коли користувач обирає пункт меню параметрів, система викликає метод **onOptionsItemSelected()** поточного Activity. Цей метод передає клас **MenuItem** обраного пункта меню. Ідентифікувати обраний пункт меню можна викликавши метод **getItemId()** (рис. 5).

```
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle item selection
    switch (item.getItemId()) {
        case R.id.new_game:
            newGame();
            return true;
        case R.id.help:
            showHelp();
            return true;
        default:
            return super.onOptionsItemSelected(item);
    }
}
```

Рис. 5

Другий метод полягає у визначенні метода, який було вказано у атрибуті **android:onClick** xml файла.

Створення контекстного меню. У контекстному меню містяться дії, які відносяться до певного елемента користувацького інтерфейса.

Традиційно існують два способи представлення контекстного меню.

- **Плаваюче контекстне меню**. Меню відображається в вигляді плаваючого списка пунктів меню коли користувач довго натискає на елемент графічного інтерфейса. Користувач може виконувати контекстну дію тільки з одним елементом інтерфейса (рис. 6).

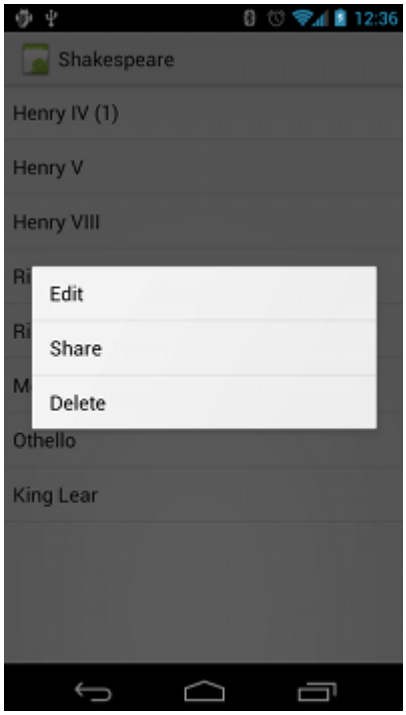

Рис. 6

- **Режим контекстних дій**. Відображається рядок контекстних дій зверху екрана з пунктами меню. Користувач може виконувати дії над декількома елементами, якщо це передбачено в додатку (рис. 7).

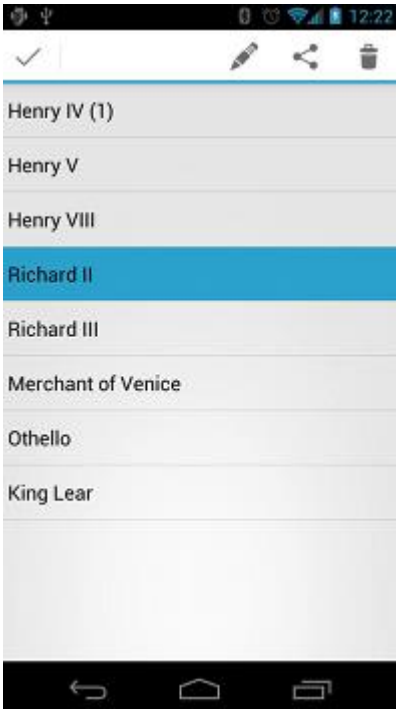

Рис. 7

Створення плаваючого контекстного меню. Для створення плаваючого контекстного меню потрібно виконати наступні дії.

1. Визначити клас View, з яким потрібно зв’язати меню. Для цього потрібно викликати метод **registerForContextMenu()** та передати йому відповідний View.
2. Реалізувати метод **onCreateContextMenu()** у відповідному Activity. Метод **MenuInflater** завантажує контекстне меню з ресурса меню (рис. 8).

```
@Override
public void onCreateContextMenu(ContextMenu menu, View v,
                                ContextMenuInfo menuInfo) {
    super.onCreateContextMenu(menu, v, menuInfo);
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.context_menu, menu);
}
```

Рис. 8

Реалізувати метод **onContextItemSelected()**. Коли користувач вибирає пункт меню, система викликає цей метод для виконання відповідної дії (рис. 9).

```
@Override
public boolean onContextItemSelected(MenuItem item) {
    AdapterContextMenuInfo info = (AdapterContextMenuInfo) item.getMenuInfo();
    switch (item.getItemId()) {
        case R.id.edit:
            editNote(info.id);
            return true;
        case R.id.delete:
            deleteNote(info.id);
            return true;
        default:
            return super.onContextItemSelected(item);
    }
}
```

Рис. 9

Використання режиму контекстних дій. При створенні додатків для Android 3.0 (рівень API 11) офіціальна документація (developer.android.com) рекомендує замість режиму плаваючого контекстного меню використовувати саме режим контекстних дій.

Коли користувач обирає елемент для взаємодії, зверху екрана відкривається рядок контекстних дій. Цей рядок містить дії, які користувач може виконати з обраним елементом. В цьому режимі користувач може обирати декілька елементів, знімати виділення та продовжити роботу з Activity. Режим контекстних дій відключається та рядок контекстних дій зникає, коли користувач знімає виділення зі всіх елементів, натискає кнопку «Назад» або обирає дію «Готово».

Використання режиму контекстних дій можливе у двох варіантах:

- Для представлення контекстних дій окремих елементів інтерфейса.
- Для пакетних контекстних дій з групами елементів в ListView або GridView.

Включення режиму контекстних дій для окремих елементів.

1. Реалізувати інтерфейс **ActionMode.Callback**. В його методах зворотного виклику можна указати дії для рядка контекстних дій, реагувати на натискання пунктів меню і т.д.
2. Викликати **startActionMode()** коли потрібно показати рядок дій.

Приклад.

1. Реалізація інтерфейсу **ActionMode.Callback**.

```
private ActionMode.Callback mActionModeCallback = new ActionMode.Callback() {

    // Called when the action mode is created; startActionMode() was called
    @Override
    public boolean onCreateActionMode(ActionMode mode, Menu menu) {
        // Inflate a menu resource providing context menu items
        MenuInflater inflater = mode.getMenuInflater();
```

```
        inflater.inflate(R.menu.context_menu, menu);
        return true;
    }

    // Called each time the action mode is shown. Always called after
onCreateActionMode, but
    // may be called multiple times if the mode is invalidated.
    @Override
    public boolean onPrepareActionMode(ActionMode mode, Menu menu) {
        return false; // Return false if nothing is done
    }

    // Called when the user selects a contextual menu item
    @Override
    public boolean onActionItemClicked(ActionMode mode, MenuItem item) {
        switch (item.getItemId()) {
            case R.id.menu_share:
                shareCurrentItem();
                mode.finish(); // Action picked, so close the CAB
                return true;
            default:
                return false;
        }
    }

    // Called when the user exits the action mode
    @Override
    public void onDestroyActionMode(ActionMode mode) {
        mActionMode = null;
    }
};
```

2. Включення режиму контекстних дій при довгому натисненні на елементі View.

```
someView.setOnLongClickListener(new View.OnLongClickListener() {
    // Called when the user long-clicks on someView
    public boolean onLongClick(View view) {
        if (mActionMode != null) {
            return false;
        }

        // Start the CAB using the ActionMode.Callback defined above
        mActionMode = getActivity().startActionMode(mActionModeCallback);
        view.setSelected(true);
        return true;
    }
});
```

Включення пакетних контекстних дій в ListView або GridView.

1. Реалізувати інтерфейс AbsListView.MultiChoiceModeListener і задати його для групи елементів View за допомогою метода **setMultiChoiceModeListener()**;
2. Викликати метод **setChoiceMode()** з аргументом **CHOICE_MODE_MULTIPLE_MODAL**.

Приклад 1.

```
ListView listView = getListView();
listView.setChoiceMode(ListView.CHOICE_MODE_MULTIPLE_MODAL);
listView.setMultiChoiceModeListener(new MultiChoiceModeListener() {

    @Override
    public void onItemCheckedStateChanged(ActionMode mode, int position,
                                          long id, boolean checked) {
        // Here you can do something when items are selected/de-selected,
        // such as update the title in the CAB
    }

    @Override
    public boolean onActionItemClicked(ActionMode mode, MenuItem item) {
        // Respond to clicks on the actions in the CAB
        switch (item.getItemId()) {
            case R.id.menu_delete:
                deleteSelectedItems();
                mode.finish(); // Action picked, so close the CAB
                return true;
            default:
                return false;
        }
    }

    @Override
    public boolean onCreateActionMode(ActionMode mode, Menu menu) {
        // Inflate the menu for the CAB
        MenuInflater inflater = mode.getMenuInflater();
        inflater.inflate(R.menu.context, menu);
        return true;
    }

    @Override
    public void onDestroyActionMode(ActionMode mode) {
        // Here you can make any necessary updates to the activity when
        // the CAB is removed. By default, selected items are
deselected/unchecked.
    }

    @Override
    public boolean onPrepareActionMode(ActionMode mode, Menu menu) {
        // Here you can perform updates to the CAB due to
        // an invalidate() request
        return false;
    }
});
```

Створення спливаючого меню. PopupMenu прив’язане до елемента View. Воно відображається нижчє елемента (рис. 10).

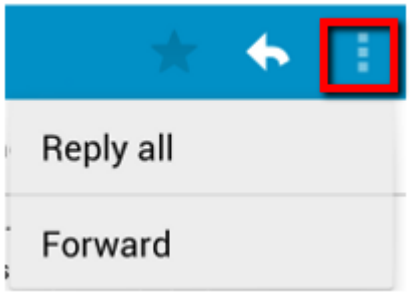

Рис. 10

Для створення меню потрібно виконати такі дії.

1. Створити екземпляр класа **PopupMenu** за допомогою конструктора, який приймає значення Context і View до яких повинно бути прив’язане меню.
2. За допомогою **MenuInflater** завантажити ресурс меню в об’єкт Menu, який повертається методом **PopupMenu.getMenu()**. Для API 14 і вище для цього можна використовувати метод **PopupMenu.inflate()**.

Приклад 2.

```
<ImageButton
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@drawable/ic_overflow_holo_dark"
    android:contentDescription="@string/descr_overflow_button"
    android:onClick="showPopup" />


public void showPopup(View v) {
    PopupMenu popup = new PopupMenu(this, v);
    MenuInflater inflater = popup.getMenuInflater();
    inflater.inflate(R.menu.actions, popup.getMenu());
    popup.show();
}
```

Обробка натиснення. Для виконання якоїсь дії при виборі конкретного пункту меню необхідно реалізувати інтерфейс **PopupMenu.OnMenuItemClickListener**.

Приклад 3.

```
public void showMenu(View v) {
    PopupMenu popup = new PopupMenu(this, v);

    // This activity implements OnMenuItemClickListener
    popup.setOnMenuItemClickListener(this);
    popup.inflate(R.menu.actions);
    popup.show();
}

@Override
public boolean onMenuItemClick(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.archive:
            archive(item);
            return true;
        case R.id.delete:
            delete(item);
            return true;
        default:
            return false;
    }
}
```

Приклад 4. Реалізація текстового редактора. Виконується читання, редагування та запис текстового файла.

xml файл activity.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <EditText
        android:id="@+id/editText"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="top|left"
        android:inputType="textMultiLine" />

</LinearLayout>
```

xml файл з пунктами меню «Open» та «Save».

```
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    tools:context=".MainActivity">

    <item
        android:id="@+id/action_open"
        android:icon="@android:drawable/ic_menu_day"
        android:orderInCategory="100"
        app:showAsAction="ifRoom|withText"
        android:title="@string/action_open" />
    <item
        android:id="@+id/action_save"
        android:icon="@android:drawable/ic_menu_save"
        android:orderInCategory="100"
        app:showAsAction="ifRoom|withText"
        android:title="@string/action_save" />
</menu>
```

Java-клас Activity.

```
package com.example.alex.editor;

import android.support.v7.app.ActionBarActivity;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.EditText;
import android.widget.Toast;
import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.io.OutputStreamWriter;

public class MainActivity extends ActionBarActivity {
    private final static String FILENAME = "sample.txt"; // имя файла
    private EditText mEditText;

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_main);

        mEditText = (EditText) findViewById(R.id.editText);
```

```
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.main_menu, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.action_open:
                openFile(FILENAME);
                return true;
            case R.id.action_save:
                saveFile(FILENAME);
                return true;
            default:
                return true;
        }
    }

    private void openFile(String fileName) {
        try {
            InputStream inputStream = openFileInput(fileName);

            if (inputStream != null) {
                InputStreamReader isr = new InputStreamReader(inputStream);
                BufferedReader reader = new BufferedReader(isr);
                String line;
                StringBuilder builder = new StringBuilder();

                while ((line = reader.readLine()) != null) {
                    builder.append(line + "\n");
                }

                inputStream.close();
                mEditText.setText(builder.toString());
            }
        } catch (Throwable t) {
            Toast.makeText(getApplicationContext(),
                    "Exception: " + t.toString(), Toast.LENGTH_LONG).show();
        }
    }

    private void saveFile(String fileName) {
        try {
            OutputStream outputStream = openFileOutput(fileName, 0);
            OutputStreamWriter osw = new OutputStreamWriter(outputStream);
            osw.write(mEditText.getText().toString());
            osw.close();
        } catch (Throwable t) {
            Toast.makeText(getApplicationContext(),
                    "Exception: " + t.toString(), Toast.LENGTH_LONG).show();
        }
    }
}
```

Завдання до лабораторної роботи

1. Протестуйте додаток з прикладу 4 та виправіть знайдені помилки. При необхідності використовуйте режим Debug.
2. Додайте у програму лабораторної роботи №5 можливість вибору налаштувань гри, використовуючи при цьому три різних типа меню.

Контрольні запитання

1. Які основні типи меню Android ви знаєте?
2. Які атрибути меню ви знаєте?
3. Яким чином оброблюється натискання пункту меню?