Тема 12. Збереження стану додатка. Пошук даних і файлів

Існує ряд випадків, коли потрібно зберігати поточний стан програми, щоб користувачі не втрачали свої дані.

Існує два типи станів, які можна зберегти: можна зберегти поточний стан інтерфейсу користувача, якщо користувач перериває введення даних, щоб він міг відновити введення при повторному запуску програми, або можна зберегти дані в більш постійному режимі. зберігати дані для доступу до нього будь-коли.

Збереження поточного стану інтерфейсу

Коли інша програма запускається і приховує програму, введені дані можуть бути не готові для збереження в постійному сховищі даних, оскільки введення ще не завершено. З іншого боку, все одно потрібно зберігати поточний стан активності, щоб користувач не втратив всю свою роботу, наприклад, під час надходження телефонного дзвінка. Зміна конфігурації пристрою Android, наприклад поворот екрана, також матиме той самий ефект, так що це ще одна вагома причина для збереження стану.

Коли відбувається одна з цих подій або інша подія, яка вимагає збереження стану дії, Android SDK викликає метод onSaveInstanceState для поточної дії, який отримує об'єкт android.os.Bundle як параметр. Якщо використовуються стандартні подання з Android SDK і ці подання мають унікальні ідентифікатори, стан цих елементів керування буде автоматично збережено в комплекті. Але якщо використовуються кілька екземплярів подання, які мають той самий ідентифікатор, наприклад, шляхом повторення подання з використанням ListView, значення, введені в елементи керування, не будуть збережені, оскільки ідентифікатор дублюється. Крім того, якщо створюються власні елементи управління, доведеться зберегти стан цих елементів управління.

Якщо потрібно вручну зберегти стан, потрібно перевизначити метод onSaveInstanceState і додати власну інформацію в отриманий android.os.Bundle як параметр з парами ключ / значення. Ця інформація буде доступна пізніше,

коли необхідно відновити дію в методах onCreate і onRestoreInstanceState. Всі типи примітивів або масиви значень цих типів можуть бути збережені у зв'язці. Якщо потрібно зберегти об'єкти або масив об'єктів у комплекті, вони повинні реалізувати інтерфейси java.io.Serializable або android.os.Parcelable.

Щоб продемонструвати збереження в стані, буде використано версію програми, яка доступна на GitHub за адресою http://github.com/CindyPotvin/RowCounter. Додаток керує лічильниками рядків, що використовуються для в'язання проекту, у цій версії користувач може створювати новий проект, і стан створюваного проекту необхідно зберегти, якщо користувач був перерваний під час створення проекту. Для демонстрації числові значення вводяться з використанням елемента керування CounterView , який не обробляє збереження стану, тому потрібно зберегти стан кожного лічильника вручну у зв'язці.

```
@Override
public void onSaveInstanceState(Bundle savedInstanceState) {
  CounterView rowCounterAmountView = (CounterView)this.findViewById(R.id.row_counters_amount);
  savedInstanceState.putInt(ROW_COUNTERS_AMOUNT_STATE, rowCounterAmountView.getValue());

  CounterView rowAmountView = (CounterView)this.findViewById(R.id.rows_amount);
  savedInstanceState.putInt(ROWS_AMOUNT_STATE, rowAmountView.getValue());

  // Call the superclass to save the state of all the other controls in the view hierarchy
  super.onSaveInstanceState(savedInstanceState);
}
```

Коли користувач повертається назад у програму, Android SDK автоматично відтворює дію з інформації, яка була збережена в пакеті. На цьому етапі також потрібно відновити стан інтерфейсу користувача для елементів управління. Можна відновити стан інтерфейсу користувача, використовуючи дані, які зберегли в комплекті з двох методів активності: метод onCreate, який

викликається першим при відтворенні дії, або метод onRestoreInstanceState, який викликається після методу onStart. Можна відновити стан в одному або іншому методі, і в більшості випадків це не має значення, але вони доступні, якщо потрібно виконати деяку ініціалізацію після методів onCreate і onStart. Ось два можливі способи відновлення стану з дії з використанням пакета, збереженого в попередньому прикладі:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
  [...Normal initialization of the activity...]
   // Check if a previously destroyed activity is being recreated.
  // If a new activity is created, the savedInstanceState will be empty
  if (savedInstanceState != null) {
    // Restore value of counters from saved state
    CounterView rowCounterAmountView;
    rowCounterAmountView = (CounterView)this.findViewById(R.id.row_counters_amount);
rowCounterAmountView.setValue(savedInstanceState.getInt(ROW_COUNTERS_AMOUNT_STATE));
    CounterView rowAmountView = (CounterView)this.findViewById(R.id.rows_amount);
rowAmountView.setValue(savedInstanceState.getInt(ROWS_AMOUNT_STATE));
  }
}
@Override
protected void onRestoreInstanceState(Bundle savedInstanceState) {
  // Call the superclass to restore the state of all the other controls in the view hierarchy
  super.onRestoreInstanceState(savedInstanceState);
  // Restore value of counters from saved stat
  CounterView rowCounterAmountView = (CounterView)this.findViewById(R.id.row_counters_amount);
rowCounterAmountView.setValue(savedInstanceState.getInt(ROW_COUNTERS_AMOUNT_STATE));
```

```
        CounterView rowAmountView =
(CounterView)this.findViewById(R.id.rows_amount);
rowAmountView.setValue(savedInstanceState.getInt(ROWS_AMOUNT_STATE));
    }
```

Збереження даних у пакеті не означає, що воно є постійним сховищем даних, оскільки воно зберігає лише поточний стан подання: воно не є частиною життєвого циклу дії та викликається лише тоді, коли необхідно відновити дію або відправлено на задній план. Це означає, що метод onSaveInstanceState не викликається, коли програма знищується, оскільки стан активності ніколи не може бути відновлено. Щоб зберегти дані, які ніколи не повинні бути втрачені, необхідно зберегти дані в одному з постійних сховищ даних.

Збереження даних у постійному сховищі даних

Якщо необхідно зберегти дані в постійному сховищі даних, коли дія відправлена у фоновий режим або знищена з будь-якої причини, необхідно зберегти дані в методі onPause дії. Метод onStop викликається лише в тому випадку, якщо інтерфейс користувача повністю прихований, тому не можна покладатися на те, що він викликається постійно. Всі необхідні дані повинні бути збережені на цьому етапі, тому що не можна контролювати, що відбудеться після: користувач може, наприклад, закрити програму, і дані будуть втрачені.

У попередній версії програми, коли користувач збільшував лічильник для рядка в проекті, програма щоразу зберігала поточне значення лічильника в базі даних. Тепер дані зберігаються лише тоді, коли користувач залишає дію, і збереження при кожному зустрічному натисканні більше не потрібно:

```
    @Override
    public void onPause() {
      super.onPause();
      ProjectsDatabaseHelper database = new ProjectsDatabaseHelper(this);
      // Update the value of all the counters of the project in the database since the activity
is
      // destroyed or send to the background
      for (RowCounter rowCounter: mRowCounters) {
```

```
    database.updateRowCounterCurrentAmount(rowCounter);
  }
```

Пізніше, якщо програма не була знищена і користувач знову отримує доступ до дії, можливі два можливі процеси в залежності від того, як ОС Android обробила дію. Якщо дія все ще знаходиться в пам'яті, наприклад, якщо користувач відкрив іншу програму і відразу ж повернувся, спочатку викликається метод onRestart, потім викликається метод onStart і, нарешті, викликається метод OnResume, і дія відображається в користувачі. Але якщо дія була повторно використана і відтворюється, наприклад, якщо користувач повернув пристрій, щоб відтворити макет, процес такий же, як і для нової дії: спочатку викликається метод onCreate, а потім викликається метод onStart і, нарешті, метод onResume, і дія показується користувачеві.

Отже, якщо потрібно використовувати дані, які були збережені в постійному сховищі даних, для ініціалізації елементів управління у діяльності, які втрачають свій стан, потрібно помістити код в метод onResume, оскільки він завжди викликається, незалежно від того, чи була дія відтворено чи ні. У попередньому прикладі немає необхідності явно відновлювати дані, оскільки ніякі елементи управління не використовувалися: якщо дія була повторно використана, вона відтворюється заново, а метод onCreate ініціалізує елементи управління з даних у базі даних. Якщо дія все ще знаходиться в пам'яті, робити більше нічого: Android SDK обробляє відображення значень у тому вигляді, в якому вони були вперше відображені, як пояснюється раніше в розділі про збереження станів інтерфейсу користувача. Ось нагадування про те, що відбувається в методі onCreate:

```
@Override
protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
  setContentView(R.layout.project_activity);
  Intent intent = getIntent();
  long projectId = intent.getLongExtra("project_id", -1);
  // Gets the database helper to access the database for the application
```

```
ProjectsDatabaseHelper database = new ProjectsDatabaseHelper(this);
// Use the helper to get the current project
Project currentProject = database.getProject(projectId);
TextView projectNameView = (TextView)findViewById(R.id.project_name);
projectNameView.setText(currentProject.getName());
    // Initialize the listview to show the row counters for the project from
    // the database
ListView rowCounterList = (ListView)findViewById(R.id.row_counter_list);
mRowCounters = currentProject.getRowCounters();
ListAdapter rowCounterListAdapter = new RowCounterAdapter(this,
                                        R.layout.rowcounter_row,
                                        currentProject.getRowCounters());
rowCounterList.setAdapter(rowCounterListAdapter);
}
```