Тема 7. Інтерфейс користувача

7.1 Компоненти інтерфейсу

Макет визначає візуальну структуру користувальницького інтерфейсу, наприклад, призначеного для користувача інтерфейсу операції або віджета додатка. Існує два способи оголосити макет:

1. Оголошення елементів призначеного для користувача інтерфейсу в XML. В Android є зручний довідник XML-елементів для класів View і їх підкласів, наприклад таких, які використовуються для віджетів і макетів.
2. Створення екземплярів елементів під час виконання. Ваша програма може програмним чином створювати об’єкти View і ViewGroup (а також керувати їх властивостями).

Платформа Android забезпечує гнучкість при використанні будь-якого з цих способів для оголошення, призначеного для користувача інтерфейсу додатка і його управління. Наприклад, можна оголосити в XML макети за замовчуванням, включаючи елементи екрана, які будуть відображатися в макетах, і їх властивості. Потім можна додати в додаток код, який дозволяє змінювати стан об’єктів на екрані (включаючи оголошені в XML) під час виконання.

Для налагодження макетів користуються інструментом Hierarchy Viewer – за його допомогою можна переглянути значення властивостей, рамки з індикаторами заповнення або полів, а також повністю прорисовані подання прямо під час налагодження програми в емуляторі або на пристрої.

За допомогою інструменту layoutopt можна швидко проаналізувати макети і їх ієрархії на предмет низької ефективності чи інших проблем.

Перевага оголошення призначеного для користувача інтерфейсу у файлі XML полягає в тому, що таким чином можна більш ефективно відокремити подання свого додатка від коду, який управляє його поведінкою. Описи призначеного для користувача інтерфейсу знаходяться за межами коду вашої

програми. Це означає, що можна змінювати або адаптувати інтерфейс без необхідності вносити правки у вихідний код і повторно компілювати його. Наприклад, можна створити різні файли XML-макета для екранів різних розмірів і різних орієнтацій екрана, а також для різних мов. Крім того, оголошення макета в XML спрощує візуалізацію структури призначеного для користувача інтерфейсу, завдяки чому налагодження проблем також стає простішим. У даному під- розділі ми навчимо вас оголошувати макет в XML. Якщо ви волієте за краще створювати екземпляри об’єктів View під час виконання, зверніться до довідкової документації для класів ViewGroup і View.

Як правило, довідник XML-елементів для оголошення елементів призначеного для користувача інтерфейсу точно відповідає структурі і правилам іменування для класів і методів – назви елементів збігаються з назвами класів, а назви атрибутів відповідають методам. Фактично, відповідність часто така точ-на, що можна з легкістю здогадатися, який атрибут XML відповідає тому чи іншому методу класу, або який клас відповідає заданому елементу XML. Однак слід зазначити, що не всі довідники є ідентичними. У деяких випадках назви можуть дещо відрізнятися. Наприклад, у елемента EditText є атрибут text, який відповідає методу EditText.setText().

2.1.1 Створення XML

За допомогою довідника XML-елементів, який є в Android, можна швидко і просто створювати макети призначеного для користувача інтерфейсу та елементи, що містяться в ньому, так само, як і при створенні веб-сторінок в HTML – за допомогою вкладених елементів.

У кожному файлі макета повинен бути всього один кореневий елемент, в якості якого повинен виступати об’єкт подання (View) або подання- групи (ViewGroup). Після визначення кореневого елемента можна приступати до додавання додаткових об’єктів макета або віджетів як дочірніх елементів для поступового формування ієрархії уявлень, яка визначає ваш макет. Нижче

показаний приклад макета XML, в якому використовується вертикальний об’єкт LinearLayout, в якому розміщені елементи TextView і Button.

2.1.2 Завантаження ресурсу XML

Під час компіляції додатка кожен файл XML макета компілюється в ресурс View. Вам необхідно завантажити ресурс макета в коді програми в ході реалізації методу зворотного виклику Activity.onCreate(). Для цього викличте метод setContentView(), передайте в нього посилання на ресурс макета в такій формі: R.layout.layout_file_name.

Метод зворотного виклику onCreate() в операції викликається платформою Android під час запуску операції.

2.1.3 Атрибути

Кожен об’єкт View і ViewGroup підтримує свої власні атрибути XML. Деякі атрибути характерні тільки для об’єкта View (наприклад, об’єкт TextView підтримує атрибут textSize), проте ці атрибути також успадкову- ються будь-якими об’єктами View, які можуть успадковувати цей клас. Деякі атрибути є загальними для всіх об’єктів View, оскільки вони успадковуються від кореневого класу View (такі, як атрибут id). Будь-які інші атрибути розглядаються як «параметри макета». Такі атрибути описують певні орієнтації макета для об’єкта View, які задані батьківським об’єктом ViewGroup такого об’єкта.

2.1.4 Ідентифікатор

Будь-який об’єкт View може бути пов’язаний з цілочисельним ідентифікатором, який служить для позначення унікальності об’єкта View в ієрархії. Під час компіляції додатка цей ідентифікатор використовується як ціле число, однак він зазвичай призначається у файлі XML-макета у вигляді рядка в атрибуті id. Цей атрибут XML є загальним для всіх об’єктів View (певним класом View), його ви будете використовувати досить часто.

Символ @ на початку рядка вказує на те, що обробнику XML слід виконати синтаксичний аналіз решти ідентифікатора і визначити його як ресурс ідентифікатора. Символ плюса (+) означає, що це ім'я нового ресурсу, який необхідно створити і додати до наших ресурсів (у файлі R.java). В Android існує ряд інших ресурсів ідентифікатора. При посиланні на ідентифікатор ресурсу Android вам не потрібно вказувати символ плюса, проте необхідно додати простір імен пакета android.

Після додавання простору імен пакета android можна послатися на ідентифікатор з класу ресурсів android.R, а не з локального класу ресурсів.

Щоб створити подання і послатися на них з програми, зазвичай слід виконати такі дії.

1. Визначити подання або віджет у файлі макета і надати йому унікальний ідентифікатор.
2. Створити екземпляр об'єкта подання і виконати його захоплення з макета (зазвичай за допомогою методу onCreate()):
3. Визначити ідентифікатори для об'єктів подання; це має важливе значення при створенні об'єкта RelativeLayout. У відносному макеті універсальні ідентифікатори використовуються для розташування уявлень щодо один одного.

Ідентифікатор не обов'язково повинен бути унікальним в рамках всієї ієрархії, а тільки в тій її частині, де ви виконуєте пошук (найчастіше це може бути якраз вся ієрархія, тому при можливості ідентифікатори повинні бути повністю унікальними).

2.1.5 Параметри макета

Атрибути макета XML, які називаються layout_something, визначають параметри макета для об'єкта подання, що підходять для класу ViewGroup, в якому він знаходиться.

Кожен клас ViewGroup реалізує вкладений клас, який успадковує ViewGroup.LayoutParams.У цьому підкласі є типи властивостей, які визначають розмір і положення кожного дочірнього подання, які підходять для його групи.

Зверніть увагу, що підклас LayoutParams має власний синтаксис для задання значень. Кожен дочірній елемент повинен визначати LayoutParams, які підходять для його батьківського елемента, тоді як він сам може визначати інші LayoutParams для своїх дочірніх елементів.

Всі групи подань включають в себе параметри ширини і висоти (layout_width і layout_height), і кожне подання має визначати їх. Багато LayoutParams також включають додаткові параметри полів і кордонів.

Для параметрів ширини і висоти можна вказати точні значення, хоча, можливо, вам не захочеться робити це часто. Зазвичай для завдання значень ширини і висоти використовується одна з таких констант:

- wrap_content – розмір подання задається за розмірами його вмісту;
- match_parent (яка до API рівня 8 називалася fill_parent) – розмір подання визначається обмеженнями, що задаються його батьківської групою подань.

Як правило, не рекомендується ставити абсолютні значення ширини і висоти макета (наприклад, у точках). Замість цього використовуйте відносні одиниці виміру, такі, як пікселі, що не залежать від дозволу екрану (dp), wrap_content або match_parent. Це гарантує однакове відображення вашого застосування на пристроях з екранами різних розмірів.

2.1.6 Розміщення макета

Подання має прямокутну форму. Розташування подання визначається його координатами зліва і зверху, а його розміри – параметрами ширини і висоти. Розташування вимірюється в пікселах.

Розташування подання можна отримати шляхом виклику методів getLeft() і getTop(). Перший повертає координату зліва (по осі X) для прямокутника подання. Другий повертає верхню координату (по осі Y) для прямокутника

подання. Обидва ці методи повертають розташування точки зору щодо його батьківського елемента. Наприклад, коли метод getLeft() повертає 20, це означає, що подання знаходиться на відстані 20 пікселів від лівого краю його безпосереднього батьківського елемента.

Крім того, є кілька зручних методів (getRight() і getBottom()), які дозволяють уникнути зайвих обчислень. Ці методи повертають координати правого і нижнього країв прямокутника подання. Наприклад, виклик методу getRight() аналогічний такому обчисленню: getLeft()+getWidth().

Розмір, відступ і поля. Розмір подання виражається його шириною і висотою. Фактично, подання має дві пари значень «ширина-висота».

Перша пара – це виміряна ширина і виміряна висота. Ці розміри визначають розмір подання в межах свого батьківського елемента. Виміряні розміри можна отримати, викликавши методи getMeasuredWidth() і getMeasuredHeight().

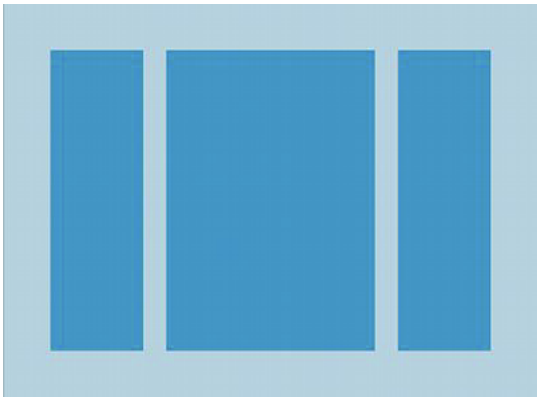
Друга пара значень – це просто ширина і висота (іноді вони називаються креслярська ширина і креслярська висота). Ці розміри визначають фактичний розмір подання на екрані після розмічування під час їх відтворення. Вони можуть відрізнятися від виміряних ширини і висоти, хоча це і не обов'язково. Значення ширини і висоти можна отримати, викликавши методи getWidth() і getHeight().

При вимірі своїх розмірів подання враховує заповнення. Відступ виражається в пікселях для лівої, верхньої, правої і нижньої частин подання. Відступ можна використовувати для зсуву вмісту подання на певну кількість пікселів. Наприклад, значення відступу зліва, що дорівнює 2, призведе до того, що вміст подання буде зміщено на 2 пікселі вправо від лівого краю подання. Для задання відступів можна використовувати метод setPadding(int, int, int, int). Щоб запросити відступ, використовують методи getPaddingLeft(), getPaddingTop(), getPaddingRight() і getPaddingBottom().

Навіть якщо подання може визначити відступ, в ньому відсутня підтримка полів. Така можливість є у групи подань.

2.1.7 Макети інтерфейсу користувача

Лінійний макет. Linear Layout (лінійний макет) – це подання групи, яке вирівнює всі дочірні елементи в одному напрямку, вертикально або горизонтально. Можна вказати напрямок макета за допомогою атрибута android:orientation.



Всі дочірні елементи LinearLayout розміщуються один за одним, таким чином, що вертикальний список має лише один дочірній елемент на кожен рядок, незалежно від його ширини, а горизонтальний список – одну висоту (висота найвищого дочірнього елемента плюс внутрішній відступ). LinearLayout розуміє зовнішні відступи між дочірніми елементами і тяжіння (праворуч, по центру або вирівнювання по лівому краю) кожного дочірнього елемента.

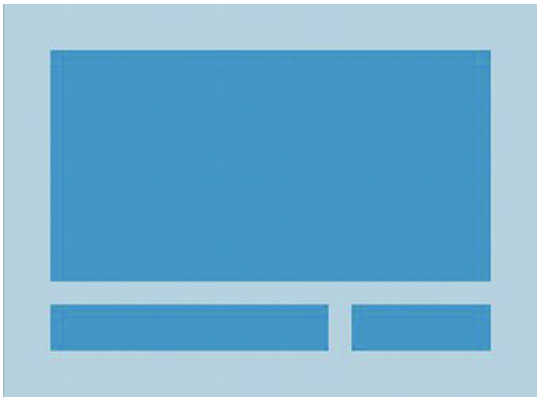
Вага макета. LinearLayout також підтримує призначення ваги окремим дочірнім елементам за допомогою атрибута android: layout_weight. Цей атрибут привласнює значення «важливості» подання з точки зору того, скільки місця воно повинне займати на екрані. Більше значення ваги дозволяє йому розширюватися, щоб заповнити все місце, що залишилось в батьківському поданні. Дочірні подання можуть точно вказувати значення ваги, і тоді все вільне місце в поданні групи призначається дочірнім елементам пропорційно до їх вказаної ваги. Значення ваги за замовчуванням дорівнює нулю.

Наприклад, якщо є три текстових поля і два з них мають вагу, що дорівнює 1, в той час як вага інших полів не вказана, то третє текстове поле без ваги не збільшиться, а займе лише область, необхідну для його вмісту. Інші два розширяться в рівній мірі, щоб заповнити місце, що залишилося після зважування всіх трьох полів. Якщо потім третьому полю призначити вагу, що

дорівнює 2 (замість 0), то його буде оголошено як більш важливе, ніж два інших, а тому воно отримує половину загального залишку місця, в той час як перші два поділяють решту порівну.

Рівноважні дочірні елементи. Для того, щоб створити лінійний макет, в якому кожен дочірній елемент займає однаковий обсяг простору на екрані, встановіть android: layout_height кожного подання таким, що дорівнює «0 dp» (для вертикального макета) або android: layout_width кожного подання таким, що дорівнює «0 dp» (для горизонтального макета). Потім встановіть android: layout_weight кожного подання таким, що дорівнює «1».

Відносний макет. RelativeLayout (відносний макет) – це подання групи, яке відображає дочірні подання у відносних позиціях. Положення кож- ного подання може бути визначене відносно споріднених елементів (наприклад, зліва від іншого подання, або нижче від нього) або положення щодо батьківсь- кого елемента області RelativeLayout, наприклад, вирівнювання по нижньо- му краю, зліва або по центру.



RelativeLayout – це дуже потужна утиліта для проектування користувацького інтерфейсу, оскільки вона може усунути вкладені подання груп і зберігати плоску ієрархію вашого макета, що підвищує продуктивність. Якщо ви використовуєте кілька вкладених груп LinearLayout, то їх можна замінити одним RelativeLayout.

Позиціонування подань. RelativeLayout дозволяє дочірнім поданням визначати їх положення щодо батьківського подання або відносно один одного (задається за допомогою ID). Таким чином, можна вирівняти два елементи за правим краєм, або розташувати один елемент нижче від іншого, розташованого

в центрі екрана, по центру зліва тощо. За замовчуванням, всі дочірні подання відмальовуються у верхньому лівому кутку макета, тому необхідно вказати положення кожного подання, використовуючи різні властивості макета, доступні з RelativeLayout.LayoutParams.

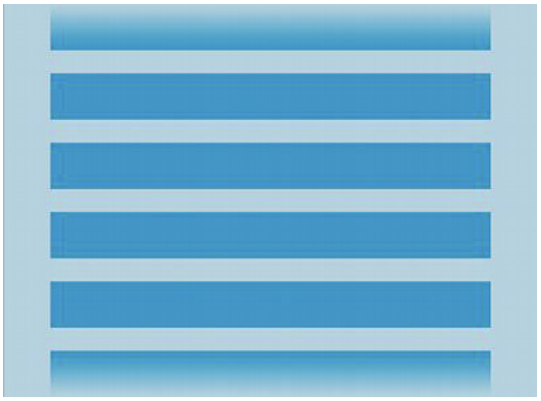
Деякі з множини властивостей макета, доступних поданням в RelativeLayout, містять у собі:

1) android:layout_alignParentTop, що розміщує верхню межу цього подання відповідно до верхньої межі батьківського елемента (якщо значення дорівнює «true»);
2) android:layout_centerVertical, що вирівнює цей дочірній еле- мент вертикально по центру усередині його батьківського елемента, якщо його значення дорівнює «true»;
3) android:layout_below, що позиціонує верхній край цього подання, розташованого нижче від подання певного ID ресурсу;
4) android:layout_toRightOf, що позиціонує лівий край цього подан- ня відповідно до правого краю подання, розташованого нижче від певного ID ресурсу.

Значення кожної властивості макета являє собою або логічний тип, щоб включати позиціонування макета відносно батьківського RelativeLayout, або ID, що посилається на інше подання у макеті, відносно якого дане подання повинне бути розташоване.

У вашому макеті XML залежності відносно інших подань у макеті можуть бути оголошені у будь-якому порядку. Наприклад, можна оголосити, що «view1» буде розташовуватися нижче від «view2», навіть якщо подання «view2» оголошено останнім у ієрархії.

Подання у вигляді списку. ListView – це подання групи, яке відображає список прокручуваних елементів. Елементи списку автоматично додаються до списку за допомогою Adapter, який витягує вміст з джерела, такого, як масив або запит бази даних, і конвертує кожен елемент у подання, поміщене у список.



Використання завантажувача. Використання CursorLoader є стандартним способом запиту Cursor як асинхронного завдання для того, щоб уникнути блокування основного потоку із запитом вашого додатка. Коли CursorLoader отримує результат Cursor, LoaderCallbacks отримує зворотний виклик до onLoadFinished(), у якому оновлюється свій Adapter з новим Cursor, і потім подання відображає результати у вигляді списку.

Хоча API-інтерфейси CursorLoader були вперше застосовані у Android 3.0 (API 11 рівня), вони також доступні у SupportLibrary, тому що ваш додаток може використовувати їх, поки пристрої, що їх підтримують, пра- цюють на Android-версії 1.6 і вище.

Більш детальну інформацію про використання Loader для асинхронного завантаження даних, див. у довіднику з Loaders.

Подання у вигляді сітки. GridView – це ViewGroup, яке відображає елементи у двовимірній сітці, що перегортується. Елементи сітки автоматично додаються до макета за допомогою ListAdapter.