05

Асинхронная обработка данных

Nikolay Moskvin
moskvin@sibext.com

31 октября 2013 г.

Содержание

Поддержка разных разрешений экранов

sibext company

- Используется два свойства: **размер** и **плотность**
- Четыре размера:
  - small
  - normal
  - large
  - xlarge
- Четыре плотности:
  - низкая (ldpi)
  - средняя (mdpi)
  - высокая (hdpi)
  - сверхвысокая (xhdpi)

Создание различных Layouts

sibext company

Из кода

```
@Override
 protected void onCreate(Bundle savedState) {
      super.onCreate(savedState);
      setContentView(R.layout.main);
}
```

Ресурсы

```
MyProject/
 res/
   layout/ <- default (portrait)
     main.xml
   layout-land/ <- landscape
     main.xml
   layout-large/ <- large (portrait)
     main.xml
   layout-large-land/ <- large landscape
     main.xml
```

Создание различных Bitmap

sibext company

Нужно использовать следующие коэффициенты для выбора scale-фактора:

- xhdpi: 2.0
- hdpi: 1.5
- mdpi: 1.0
- ldpi: 0.75

```
MyProject/
 res/
 drawable-xhdpi/ <- 200x200
      awesomeimage.png
 drawable-hdpi/ <- 150x150
      awesomeimage.png
 drawable-mdpi/ <- 100x100
      awesomeimage.png
 drawable-ldpi/ <- 75x75 (deprecated)
      awesomeimage.png
```

Поддержка разных языков

sibext company

/values-en/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string name="title">The best application</string>
</resources>
```

/values-fr/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string name="title">La meilleure application</string>
</resources>
```

Использование

```
String title = getResources().getString(R.string.title);

TextView textView = new TextView(this);
textView.setText(R.string.title);

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/title" />
```

Ресурсы

```
MyProject/
 res/
   values/
     strings.xml
   values-en/
     strings.xml
   values-fr/
     strings.xml
```

Поддержка разных платформ

sibext company

Из кода

```
if (Build.VERSION.SDK_INT
        >= Build.VERSION_CODES.HONEYCOMB) {
// add your logic...
}
```

Подробно о версиях: http://developer.android.com/reference/android/os/Build.VERSION_CODES.html

Манифест

```
<manifest ... >
    <uses-sdk android:minSdkVersion="8"
        android:targetSdkVersion="18" />
    ...
</manifest>
```

Поддержка разных платформ

sibext company

Мы можем задать параметры для Layouts из кода:

```
RelativeLayout.LayoutParams params =
  new RelativeLayout.LayoutParams(
    RelativeLayout.LayoutParams.WRAP_CONTENT,
    RelativeLayout.LayoutParams.WRAP_CONTENT);
params.addRule(
  RelativeLayout.ALIGN_PARENT_LEFT,
  RelativeLayout.TRUE);
Button button1 = new Button();
button1.setLayoutParams(params);

params = new RelativeLayout.LayoutParams(
    RelativeLayout.LayoutParams.WRAP_CONTENT,
    RelativeLayout.LayoutParams.WRAP_CONTENT);
params.addRule(RelativeLayout.RIGHT_OF,
    button1.getId());
Button button2 = ...;
button2.setLayoutParams(params);
```

Жизненный цикл процессов

sibext company

Имеется пять уровней важности:

- Foreground process (высокий приоритет, убивается в последнюю очередь)
- Visible process (обычно переходит в это состояние по onPause())
- Service process (используется для проигрывания музыки, выкачивания данных из интернета)
- Background process (Убивается по принципу наиболее давно используемое)
- Empty process (часто убиваются, используются обычно для кэширования)

Thread

sibext company

- Thread – основа для параллельного выполнения
- Каждый поток имеет свой собственный стэк вызовов и локальных переменных
- Приложение содержит как минимум один поток – “main”

Многопоточность

sibext company

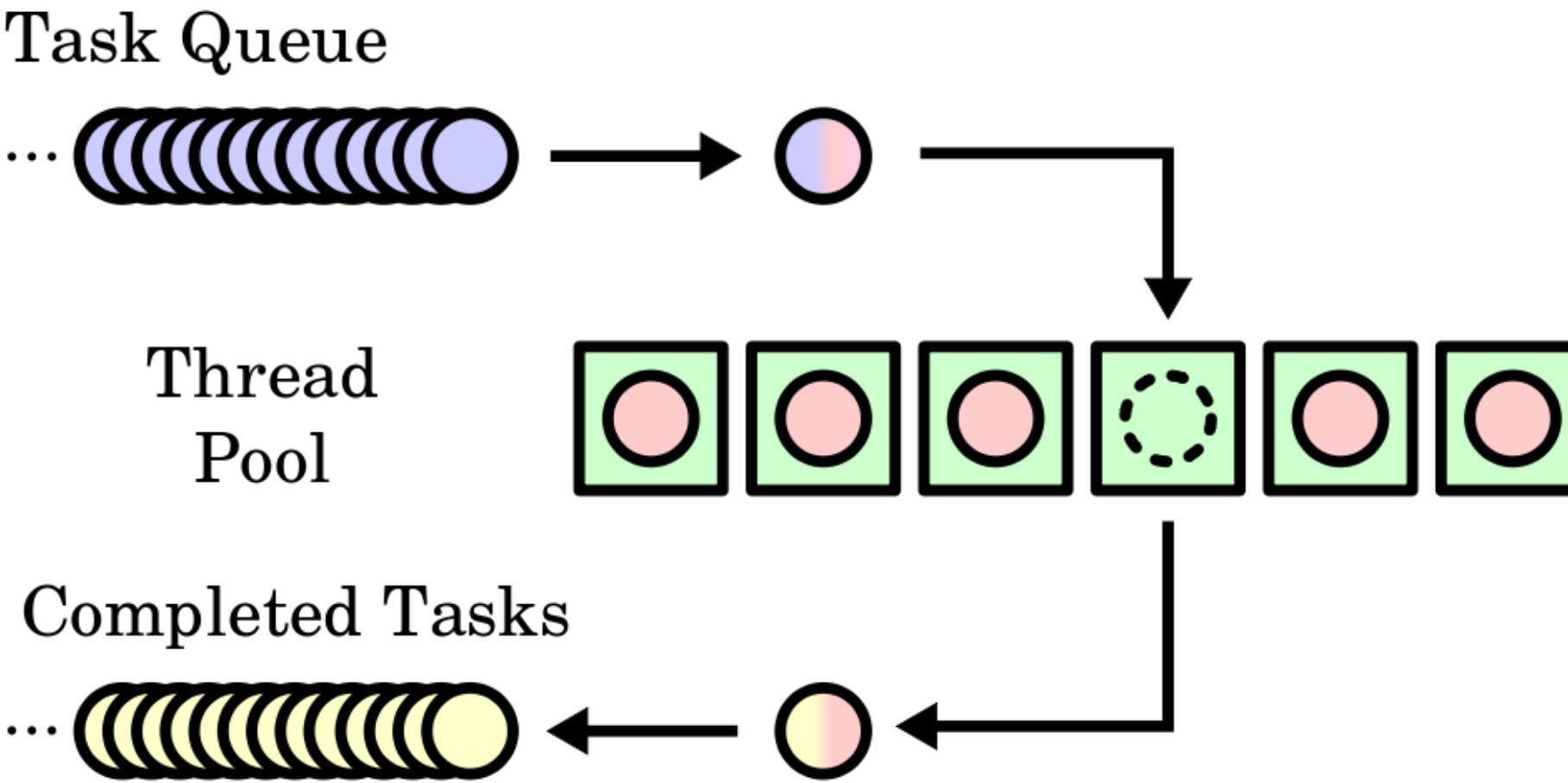
- **Handler** – для выполнения отложенных операций (worker thread)
- **Activity** – runOnUiThread
- **View** – post, postDelayed

Пул потоков

sibext company

- В Java требует достаточно большого кол-ва ресурсов
- Большое кол-во потоков отрицательно сказывается на производительности
- Решение есть – использовать шаблон “Пул потоков”

Пул потоков



AsyncTask

sibext company

- Удобный инструмент
- Поддержка с Android 1.5
- Основано на пуле потоков
- Основная работа в методе doInBackground
- `http://developer.android.com/reference/android/os/AsyncTask.html`

AsyncTask – Генерики

sibext company

- Три параметра
  - Parameter – передается при старте
  - Progress – промежуточный результат
  - Result – результат выполнения
- Можно воспользоваться классом **Void**, если не нужны параметры

AsyncTask – Запуск

sibext company

- Экземпляр класса AsyncTask должен быть создан в “main”
- execute(Param. . . params) – запуск AsynkTask
- Нельзя вызывать onPreExecute, onPostExecute, doInBackground и onProgressUpdate в ручную
- Один экземпляр может быть выполнен только один раз!

AsyncTask – Завершение

sibext company

- **cancel** – останавливает задачу в любой момент
- После завершения doInBackground будет вызван onCanceled, а не onPostExecute
- isCanceled будет возвращать true
- Необходимо добавлять проверку флага в onPostExecute

Играем в Волейбол

sibext company

- Автоматически составляет все сетевые запросы
- Обеспечивает прозрачность дискового кэширования и кэширования в памяти.
- Присутствует инструмент отладки и трассировки
- `https://android.googlesource.com/platform/frameworks/volley`

Подключаем Волейбол

sibext company

Делаем изменение в вашем pom.xml

```
<dependencies>
...
<dependency>
  <groupId>com.google</groupId>
  <artifactId>volley</artifactId>
  <version>1.0.0-SNAPSHOT</version>
</dependency>
...
</dependencies>
...
<repositories>
  <repository>
    <id>Sibext repository</id>
    <url>http://mvn.sibext.com/</url>
  </repository>
<repositories>
...
```

```
ImageLoader imageLoader =
  new ImageLoader(
    Volley.newRequestQueue(this),
    new BitmapLruCache());
imageLoader.get("http://sibext.com/i/1.png",
  new ImageLoader.ImageListener() {
  @Override
  public void onResponse(
      ImageLoader.ImageContainer imageContainer,
      boolean b) {
    // TODO: image was loaded
  }

  @Override
  public void onErrorResponse(
    VolleyError volleyError) {
    // TODO: Please inform about failed
  }
});
```

В следующей лекции

sibext company

- Подробно разберем Android манифест
- Рассмотрим понятия сервисов и провайдеров
- Узнаем каким образом можно обмениваться между Service и Activity