Министерство образования Новосибирской области

ГБПОУ НСО «Новосибирский авиационный технический колледж имени Б.С.Галущака»

**Добавление библиотек к проекту**

Самостоятельная работа 5

Междисциплинарный курс: МДК.01.03Разработка мобильных приложений

Специальность: 09.02.07 Информационные системы и программирование

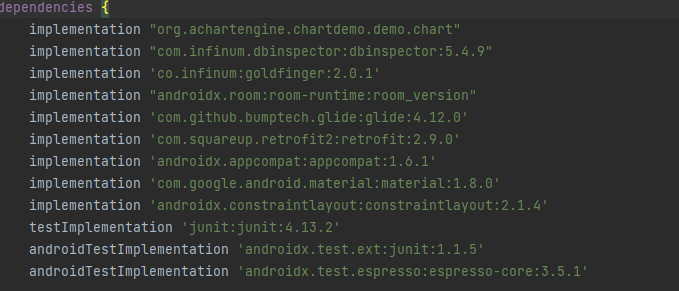
Разработал:

И.С. Климова

Выполнил: А.А Патрин

2023

Подключенные библиотеки



Контрольные вопросы

1.

Библиотека (от англ. library) в программировании - сборник подпрограмм или объектов, используемых для разработки программного обеспечения (ПО). Для ОС Android существует большое количество подключаемых библиотек. Их можно классифицировать в зависимости от их предназначения. Выделим следующие группы:

* Библиотеки совместимости. Они позволяют использовать возможности, появившиеся в какой-то версии ОС Android, на более ранних версиях платформы. Дело в том, что новые версии API выходят гораздо быстрее, чем в широком использовании оказываются устройства, поддерживающие эту версию. Разработчик с одной стороны должен ориентироваться на новые возможности и уметь их использовать, а с другой - стараться сделать так, чтобы приложение работало на максимальном количестве устройств. Библиотеки совместимости позволяют сделать это противоречие менее жестким.
* Библиотеки специального назначения. Используются для разработки игр, работы с социальными сетями, сбора статистики и в других случаях.
* Библиотеки, предоставляющие дополнительные возможности. В эту категорию можно отнести большое количество самых разных библиотек. Сюда можно отнести библиотеки рисования графиков, работы с изображениями, модифицированные элементы управления и многое другое.

2. Обеспечение безопасности осуществляется на двух уровнях. Первый - непосредственно в библиотеке, где может быть служба безопасности (в крупных библиотеках), пожарный надзор. Их основным предназначением является постоянная профессиональная *деятельность* *по* обеспечению безопасности библиотеки. Весь персонал и *подразделения* библиотеки обязаны в рамках своих должностных инструкций способствовать обеспечению безопасности.

Второй уровень обеспечивают внешние органы и организации, которые функционируют самостоятельно и не подчиняются руководству библиотеки, но *деятельность* которых оказывает существенное влияние на *безопасность* предприятия.

3.

Рассмотрим пример подключения библиотеки AChartEngine, предназначенной для построения графиков. На сайте разработчика помимо самой библиотеки можно найти подробную документацию, оформленную в стиле Javadoc pages, примеры использования библиотеки, а также ее исходный код.

В разделе Downloads также можно скачать демонстрационные примеры использования библиотеки.

Создайте проект Graphics.

Чтобы подключить библиотеку, нужно скачать ее с сайта разработчика, потом просто перетащить из проводника в папку libs вашего проекта в Eclipce.

Теперь нужно добавить строку <activity android:name="org.achartengine.GraphicalActivity"/> в раздел <application> в файле манифеста вашего проекта. Библиотека подключена.

Теперь перейдем к файлу MainActivity.java. Создайте и инициализируйте три массива (в методе onCreate()). В первом будут содержаться цвета, во втором - подписи, в третьем - значения.

int[] values = new int[] { 25,25,25,25 };

String[] bars = new String[] {"Bananas", "Kiwi", "Oranges", "Cream"};

int[] colors = new int[] { Color.YELLOW, Color.GREEN, Color.RED, Color.WHITE };

Создайте объект CategorySeries и загрузите в него массивы строк с подписями и значения. При построении диаграмм будут использоваться именно эти данные.

CategorySeries series = new CategorySeries("Pie Chart");

Создайте объект DefaultRenderer. CategorySeries и DefaultRenderer определены в библиотеке AChartEngine.

DefaultRenderer dr = new DefaultRenderer();

Для каждого цвета в диаграмме создайте SimpleSeriesRenderer и установите его в этот цвет. Затем каждый SimpleSeriesRenderer добавьте в DefaultRenderer. Таким образом, имеем один Render на каждый Series. К слову, Render может включать другой Render, таким образом можно создавать подкатегории.

for (int v=0; v<4; v++){

series.add(bars[v], values[v]);

SimpleSeriesRenderer r = new SimpleSeriesRenderer();

r.setColor(colors[v]);

dr.addSeriesRenderer(r);

}

Создайте объект PieChartIntent и передайте ему CategorySeries и DefaultRenderer.

return ChartFactory.getPieChartIntent(this, series, dr, "Fruit Salad");

С помощью PieChartIntent можно показать диаграмму, как отдельную деятельность.

С помощью методов библиотеки можно изменить размер и цвет подписей, включить зум:

dr.setZoomEnabled(true);

dr.setChartTitleTextSize(20);

dr.setLegendTextSize(TEXT\_SIZE);

dr.setChartTitleTextSize(20);

dr.setZoomButtonsVisible(false);

dr.setLabelsTextSize(TEXT\_SIZE);

dr.setLegendTextSize(TEXT\_SIZE);

dr.setLabelsColor(Color.BLACK);

Для удобства можно создать дополнительную переменную, в которой будет храниться размер текста.

private static final int TEXT\_SIZE = 40;

Intent intent = buildIntent();

startActivity(intent);

}

public Intent buildIntent() {

int[] values = new int[] { 25,25,25,25 };

. . .

dr.addSeriesRenderer(r);

}

dr.setChartTitleTextSize(20);

. . .

dr.setLabelsColor(Color.BLACK);

return ChartFactory.getPieChartIntent(

this, series, dr, "Fruit Salad");

}