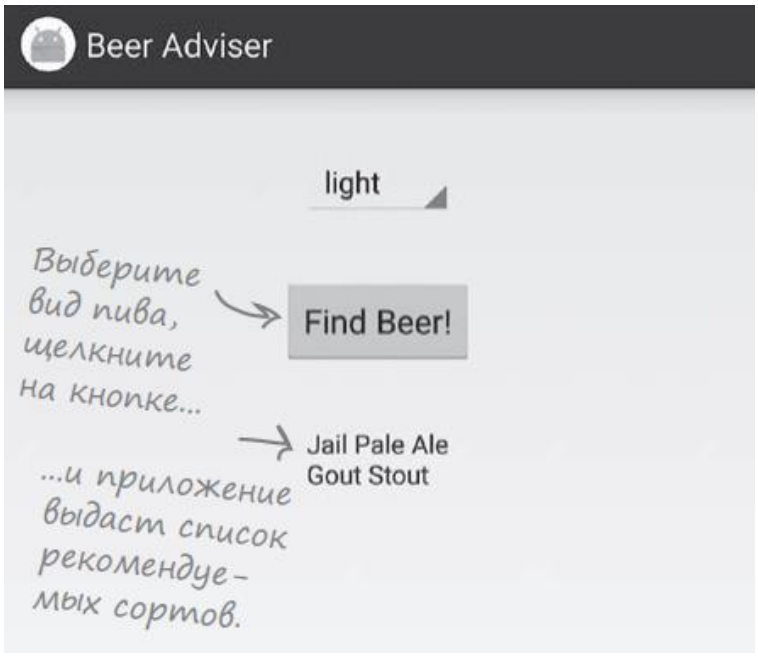
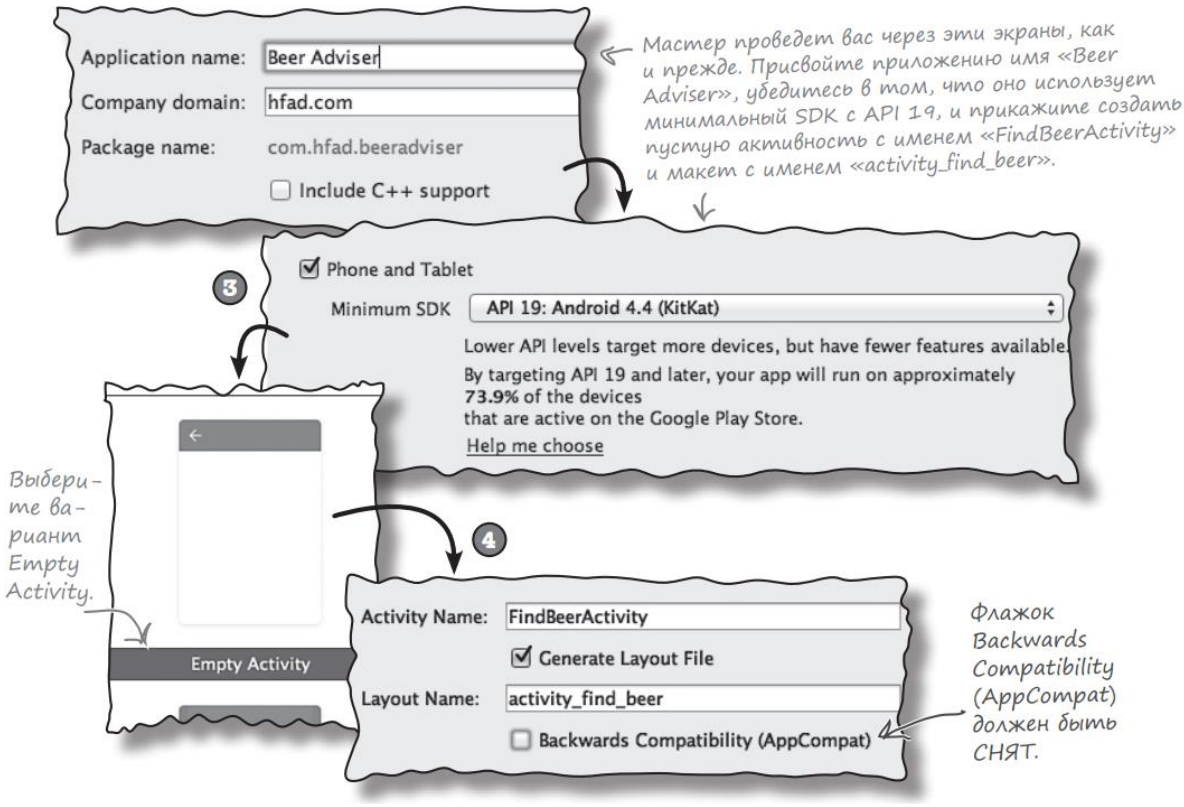
## Лабораторная работа 1

## Создание простейшего приложения.

**Задание:** создать простейшее приложение для выбора пива, в котором пользователь выбирает вид пива, который он предпочитает, щелкает на кнопке и получает список рекомендуемых сортов.




1. Создайте новый модуль с именем «`Beer Adviser`».



2. Создание интерфейса приложения:

Приложение состоит из трех компонентов графического интерфейса:

• раскрывающегося списка значений, в котором пользователь выбирает нужный вид пива;

• кнопки, которая при нажатии возвращает подборку сортов пива;

• надписи для вывода сортов пива.

Используйте контейнер <LinearLayout> для вывода компонентов графического интерфейса рядом друг с другом, по вертикали.

В режиме дизайнера добавьте необходимые компоненты на экран. Выпадающий список имеет название spinner.

Измените названия компонентов (id): TextView – “brands”; spinner – “color”.

Спозиционируйте элементы так, чтобы кнопка находилась по центру экрана, а выпадающий список и надпись на одинаковом расстоянии от кнопки. Воспользуйтесь для этого свойствами gravity и margin. Обратите внимание, что элементы должны вмещать в себя весь текст, который на них выводится.

3. Использование строкового ресурса в макете.

Создайте два строковых ресурса: для текста, выводимого на кнопке, и для текста, выводимого на надписи по умолчанию. Для этого найдите на панели Android Studio файл strings.xml в папке app/src/main/res/values. Сделайте на нем двойной щелчок, чтобы открыть его.

Добавьте новые ресурсы с именем "find_beer" и "brands":

```
<resources>
    <string name="app_name">Beer Adviser</string>
    <string name="find_beer">Find Beer!</string>
    <string name="brands">No beers selected</string>
</resources>
```

← Текст по умолчанию для надписи.

Измените элементы кнопки и надписи в разметке XML макета, чтобы в них использовались два только что добавленных строковых ресурса.

Вывести текст... → android:text="@string/find_beer" /> ← ...строкового ресурса find_beer.

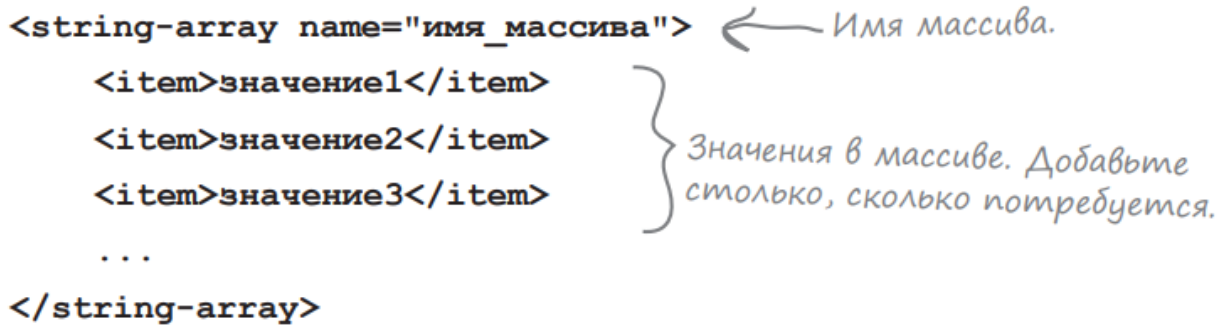
Запись @string приказывает Android найти текстовое значение в файле строковых ресурсов. Вторая часть, find_beer, приказывает Android получить значение ресурса с именем find_beer.

*! Android Studio иногда выводит в редакторе кода значения ссылок вместо кода. Например, редактор может вывести текст "Find Beer!" вместо настоящего кода "@string/find_beer". Все такие замены должны подсвечиваться в редакторе кода. Если щелкнуть на них или навести на них указатель мыши, откроется реальный код.*

4. Добавление значений в список

Создайте ресурс списка значений для раскрывающегося списка. Для этого нужно определить массив строковых значений и передать ссылку на него раскрывающемуся списку.

Синтаксис добавления массива строк выглядит так:




```
<string-array name="имя_массива">
    <item>значение1</item>
    <item>значение2</item>
    <item>значение3</item>
    ...
</string-array>
```

Добавьте следующие значения в массив: `light, amber, brown, dark`.

Для обращения к массиву строк в макете используется конструкция:

`"@array/имя_массива"`

Теперь выведите список значений в раскрывающемся списке:

`android:entries="@array/beer_colors"`

5. Добавить обработчик события для кнопки.

Чтобы щелчок на кнопке приводил к вызову метода активности, необходимо внести изменения в двух файлах:

- Изменения в файле макета. Необходимо указать, какой метод активности должен вызываться при щелчке на кнопке.

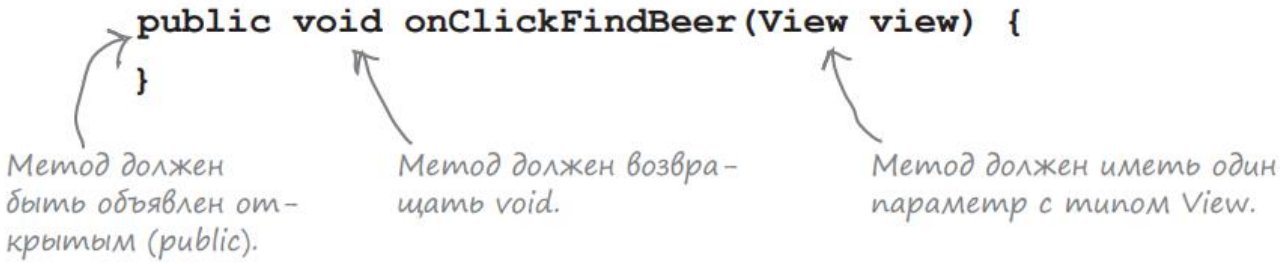
для этого нужно — добавить атрибут `android:onClick` в элемент и указать имя вызываемого метода:

`android:onClick="method_name"`

Задайте имя метода: `onClickFindBeer().`

- Изменения в файле активности `FindBeerActivity.java`. Необходимо написать метод, который будет вызываться при щелчке

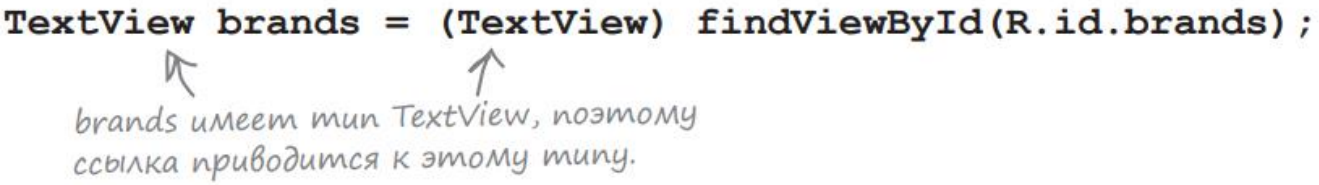
Метод `onClickFindBeer()` должен иметь строго определенную сигнатуру; в противном случае он не будет вызываться при щелчке на кнопке, указанной в макете. Он имеет следующую форму:




6. Добавление логики.

Приложение должно выводить подборку сортов пива, соответствующих виду, выбранному пользователем. Для этого необходимо сначала получить ссылки на оба компонента графического интерфейса в макете — раскрывающийся список и надпись. С помощью этих ссылок мы сможем получить значение, выбранное в списке (вид пива), и вывести текст в надписи.

Для получения ссылки на компонент графического интерфейса можно воспользоваться методом `findViewById()`. Метод `findViewById()` получает идентификатор компонента в виде параметра и возвращает объект `View`. Далее остается привести возвращаемое значение к правильному типу компонента (например, `TextView` или `Button`).




Теперь вы можете вызывать его методы.

Для задания свойства `text` используется метод `setText()`.

Также получите ссылку на раскрывающийся список; это делается практически так же, как для надписи. Снова используется метод `findViewById`(), но на этот раз результат приводится к типу `Spinner`. Зададим имя ссылке например «`color`».

Для получения текущего выбранного варианта в списке используется метод `getSelectedItem()`.

Далее его надо преобразовать к типу `String`. Дело в том, что значения раскрывающегося списка не обязаны быть строковыми объектами — это могут быть, например, изображения. Метод `String.valueOf()` используется для преобразования выбранного варианта из `Object` в `String`.

- Напишите метод `onClickFindBeer()` сами или используя подсказки в конце лабораторной.
- Протестируйте приложение. Убедитесь в том, что выбранный вариант правильно читается из списка.

```
public void onClickFindBeer(_____view) {

    //Получить ссылку на TextView
    _____brands = _____ _____(_____);

    //Получить ссылку на Spinner
    Spinner_____= _____ _____(_____);

    //Получить вариант, выбранный в Spinner
    String _____= String.valueOf(color._____);

    //Вывести выбранный вариант
    brands._____(beerType);
}
```

7. Построение вспомогательного класса

Добавьте класс `BeerExpert` в свой проект. Выделите имя пакета в папке `app/src/main/java` и выполните команду `File → New → Java Class.`

Присвойте файлу имя «`BeerExpert`» и убедитесь в том, что пакету присвоено правильное имя. Команда создает файл `BeerExpert.java`.

Добавьте в класс следующий код:

```
import java.util.ArrayList;
import java.util.List;

public class BeerExpert {

    public BeerExpert() {
    }

    List<String> getBrands(String color) {
        List<String> brands;
        brands = new ArrayList<>();
        if (color.equals("amber")) {
            brands.add("Jack Amber");
            brands.add("Red Moose");
        } else {
            brands.add("Jail Pale Ale");
            brands.add("Gout Stout");
        }
        return brands;    }}
```

8. Теперь доработайте метод `onClickFindBeer()`, чтобы он вызывал метод класса BeerExpert для получения рекомендаций.

- Объявите в классе `Activity` переменную `expert` класса `BeerExpert.`
- В метод `onClickFindBeer()` допишите:

```
String tmp;
List<String>brandsList = expert.getBrands(beerType);
for (int i =0;i<brandsList.toArray().length;i++) {
    tmp+=brandsList.get(i);
    tmp+='\n';
```

← Получить контейнер List с сортами пива.

← Построить String по данным из List.

← Каждый сорт выводится с новой строки.

- Теперь сделайте, чтобы компонент `brands` выводил полученный список рекомендуемых сортов пива.

Протестируйте приложение.

Подсказки ☺


getSelectedItem()
beerType
(Spinner)
findViewById
Button
findView
View
R.view.color
(TextView)
findView
R.id.brands
color
TextView
R.view.brands
findViewById
setText
R.id.color