

Android Sizes

Layout_Height

Layout_Width

Are mandatory
attributes in
XML

Android needs
to know how
big you want
to draw things.

Wrap
Content

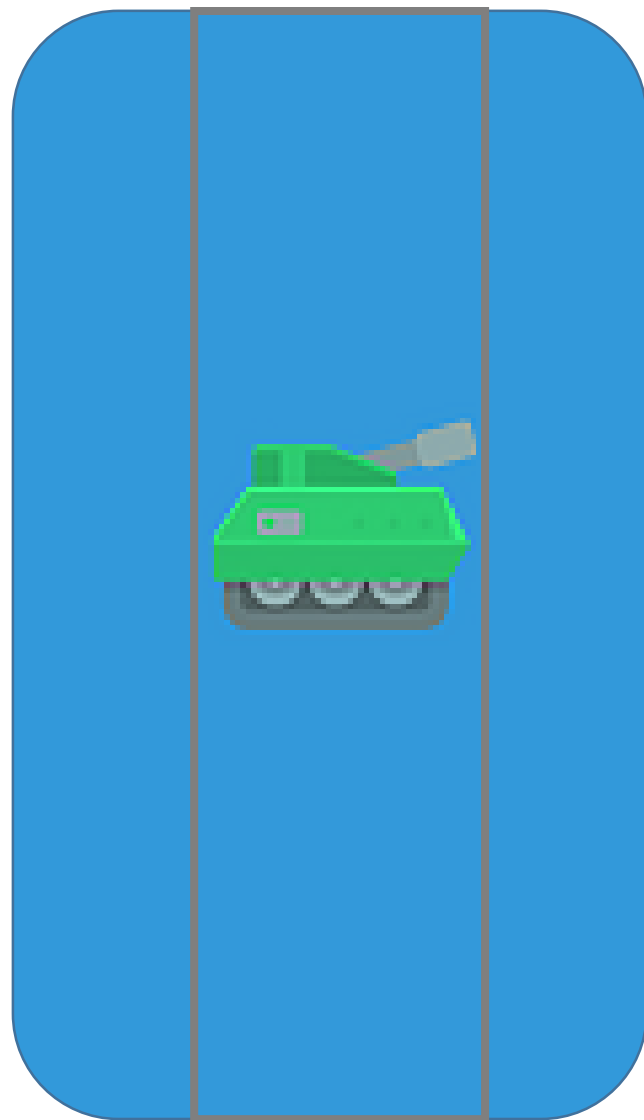
WORD

Width = Wrap Content
Height = Wrap Content

Match
Parent

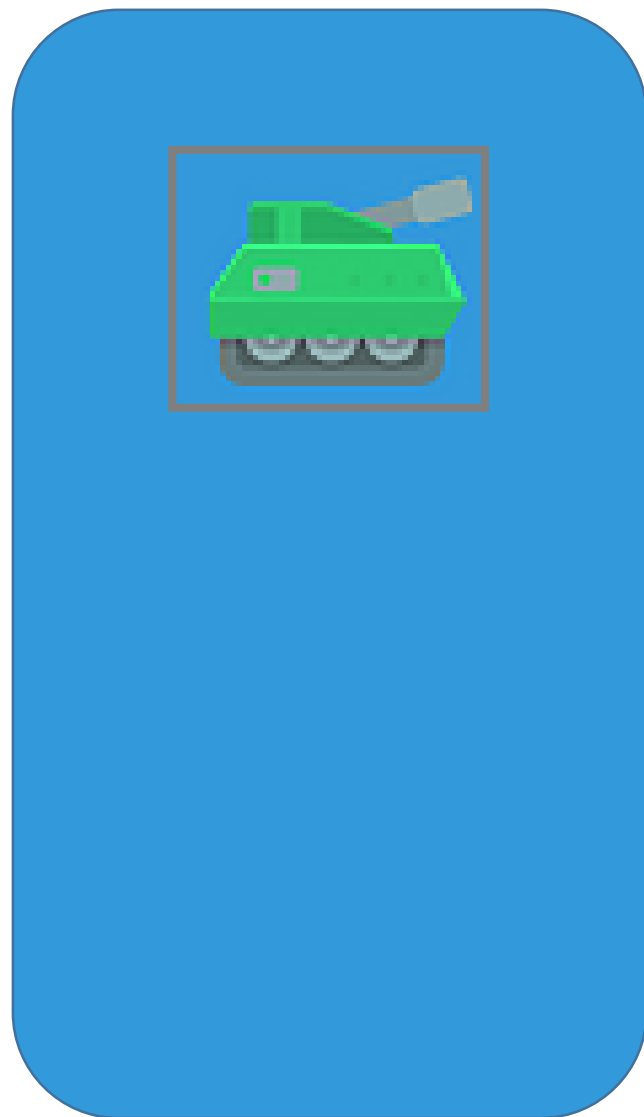
WORD

Width = Match Parent
Height = Wrap Content



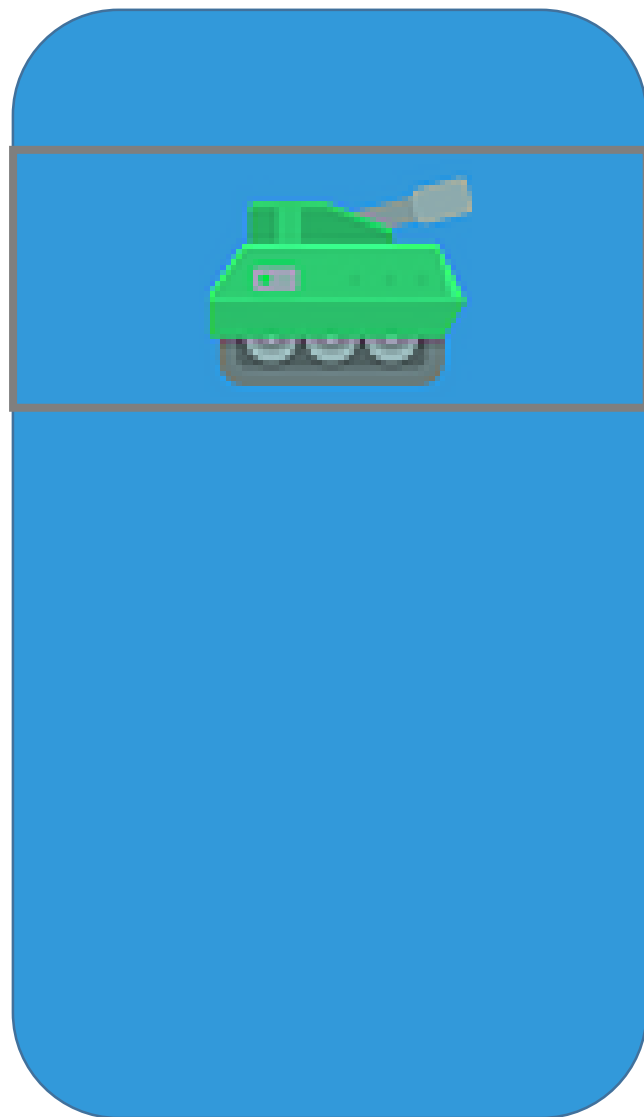
Match_parent or
wrap_content?

```
<ImageView  
    android:src="@drawable/tank"  
    android:layout_width="_____"  
    android:layout_height="_____"  
>
```



Match_parent or
wrap_content?

```
<ImageView  
    android:src="@drawable/tank"  
    android:layout_width="_____"  
    android:layout_height="_____"  
>
```



Match_parent or
wrap_content?

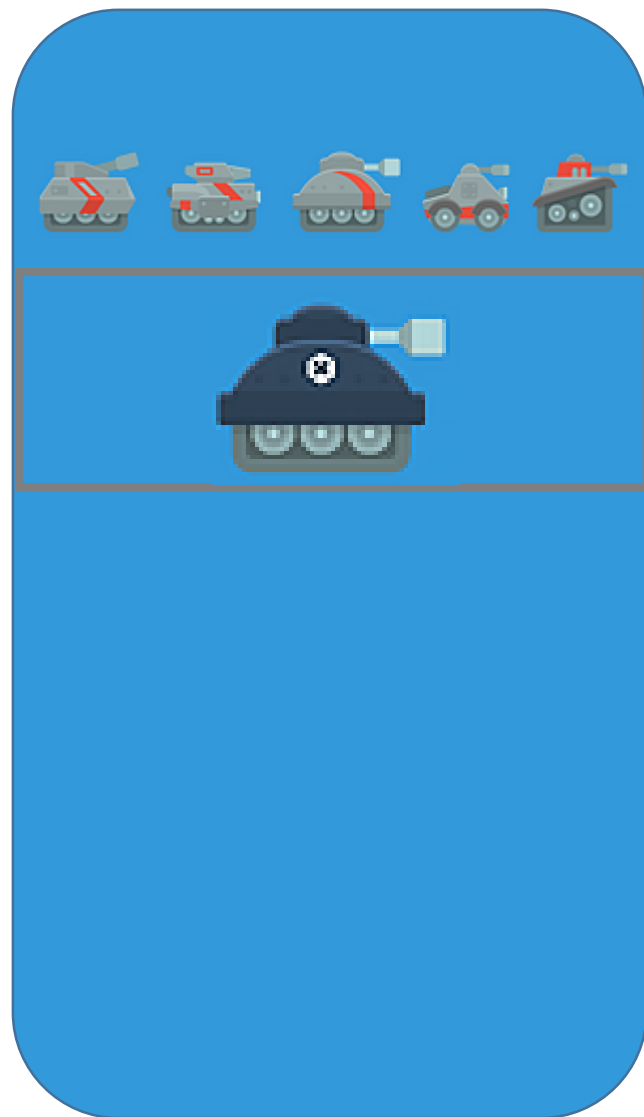
```
<ImageView  
    android:src="@drawable/tank"  
    android:layout_width="_____"  
    android:layout_height="_____"  
>
```



Match_parent or
wrap_content?

```
<ImageView  
    android:src="@drawable/tank"  
    android:layout_width="_____"  
    android:layout_height="_____"  
>
```

A common
error



These 5
images?

This
image?

Match_parent or
wrap_content?

```
<ImageView  
    android:src="@drawable/tank"  
    android:layout_width="____"  
    android:layout_height="____"  
/>
```


DIFFERENT WIDTHS ON CHILD VIEWS

200 dp

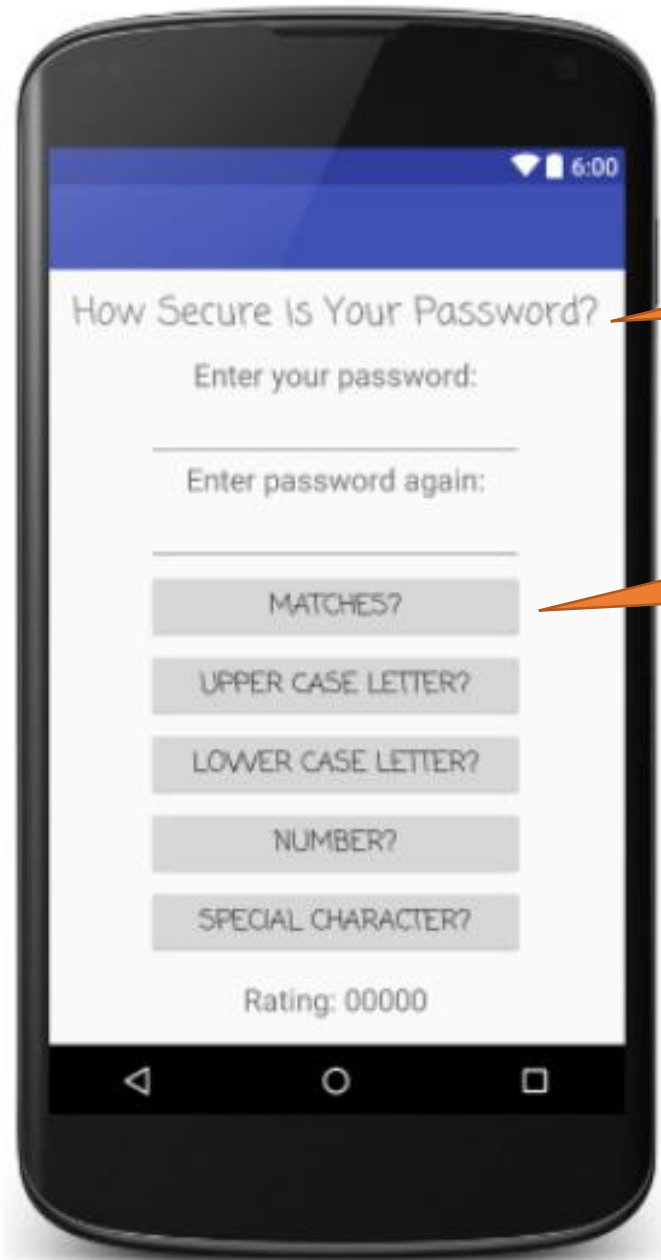


wrap-content



match-parent

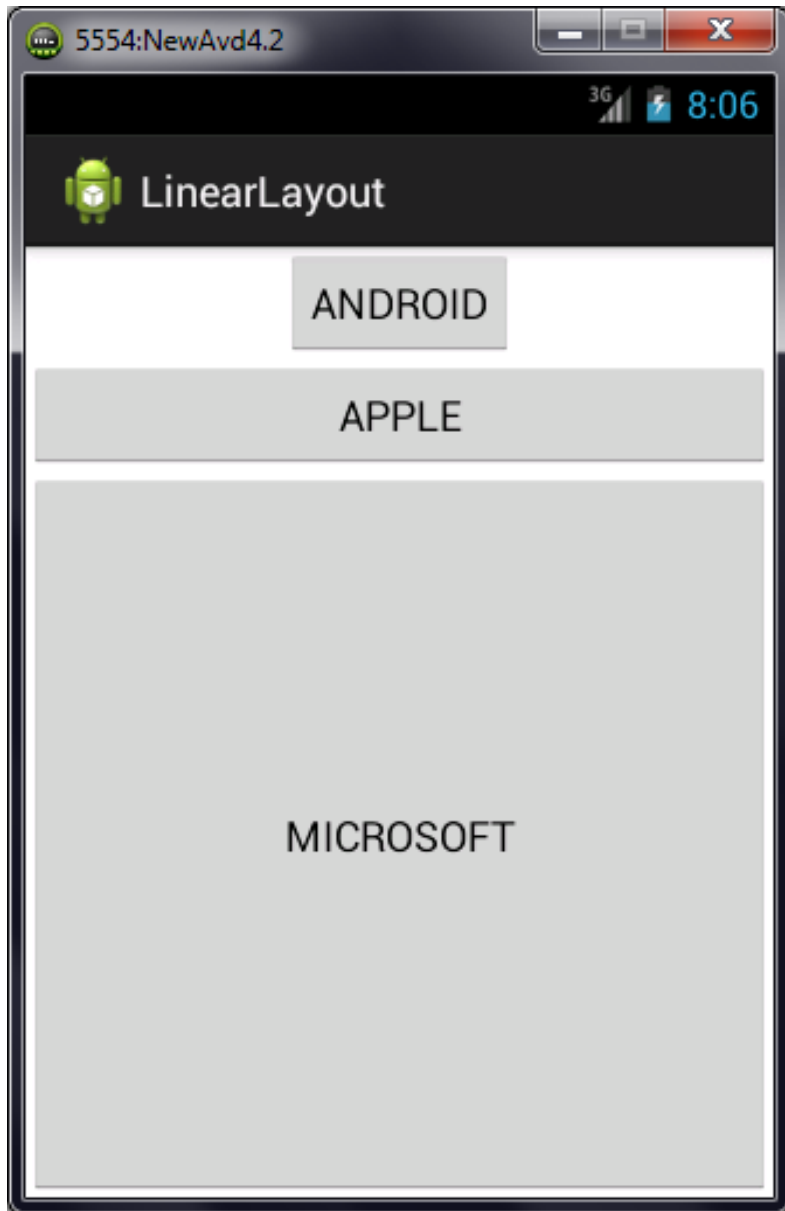




Width &
Height?

Width &
Height?

If a series of Views are all formatted the same, their height and width use DP, not wrap content.

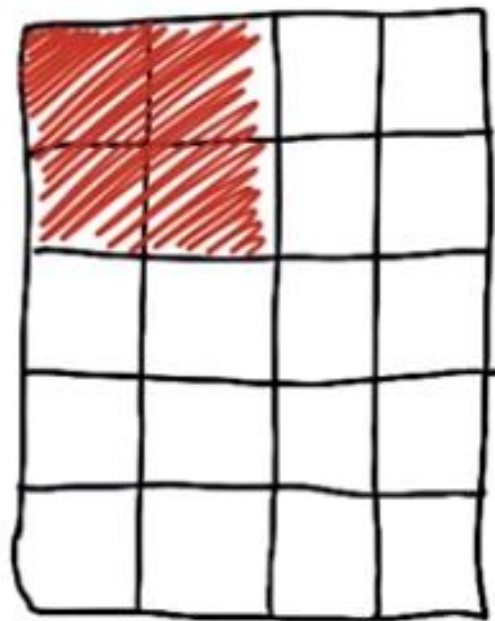


Wrap Content or
Match Parent or
Fixed DP?

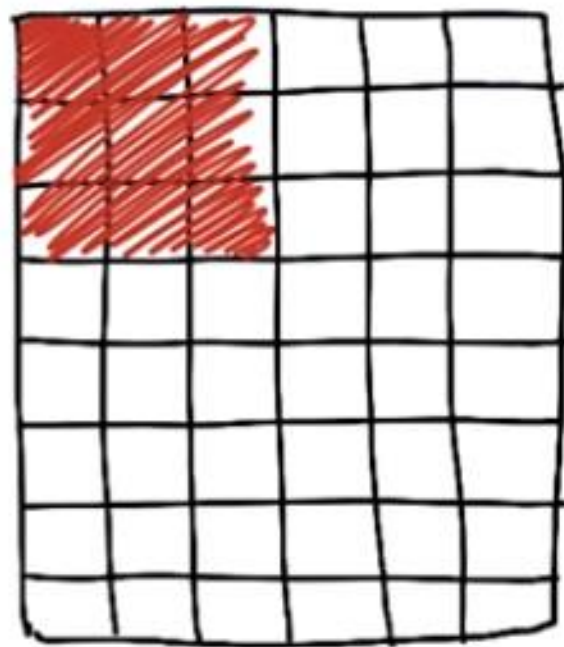


DENSITY - INDEPENDENT PIXELS

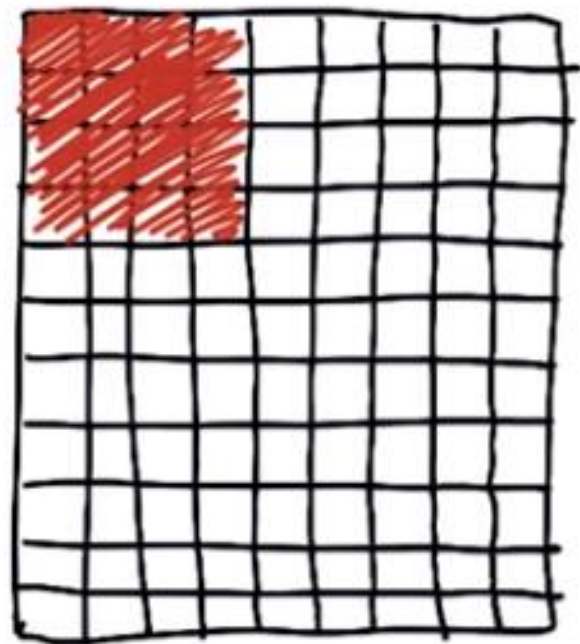
2 dp by 2 dp



Medium Resolution Device



High Resolution Device



Extra-High Resolution Device

Density Independent Pixels (DP)

- This is a size in Android. It can be used for height, width, text, margins, padding.
- It isn't a fixed size. It is re-calculated for different phone screen resolutions.
- This means that the views are a standard proportion of the screen and don't get smaller as screen resolutions improve.

DESCRIPTION

SAMPLE CODE



```
<TextView
```

```
    android:layout_width="160dp"
```

```
    android:layout_height="80dp"
```

```
    android:background="#00FF00"
```

```
    android:text="Hello"/>
```

The screen of an **Android device** is made of rows and columns of glowing dots called **pixels**. Devices can range in **screen density**, which means how many pixels per inch (or dots per inch) are on the screen. For example, an mdpi (or medium density device) has 160 dots per inch, while an xxhdpi (extra extra high density device) has 480 dots per inch.

If we specify the size of views in pixel values, then views would appear very small on the higher density devices, where there are many pixels packed into a small area. If a button is too small, then it would be hard for the user to touch it.

To achieve a consistent physical size of Views, across devices of different screen densities, we use a unit of measure called a **density-independent pixel** (**dp** or **dip**, pronounced "dee pee" or "dip"). 1 dp is equal to 1 pixel on an mdpi device. 1 dp is equal to 3 pixels on an xxhdpi device, and so on [for other devices](#). Per Material design guidelines, any touch target on the screen should be [at least 48dp wide by 48dp tall](#). That way, a button in an app on one device will be approximately the same physical size as that button in the same app running on a device with a different screen density.

GETTING PAST ERRORS

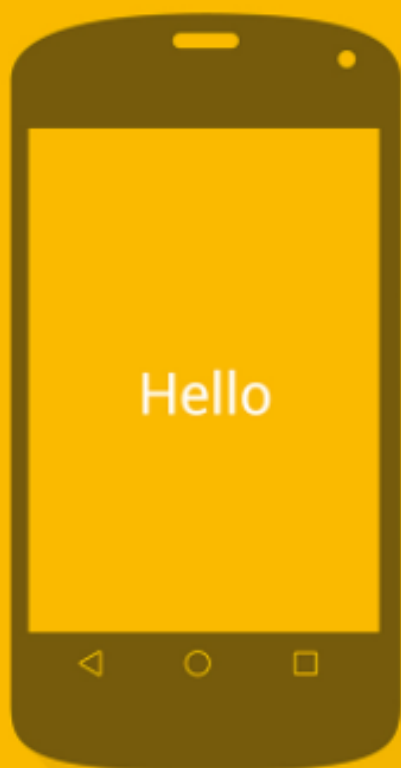
```
<Text View
  android:text="Hapy Birthday"
  android:layout_width="150dp"
  android:layout_height="150"
  android:background="@android:color/darker_groy"
>
```

Uh oh! Invalid
XML... help!

4 errors.

WRAP_CONTENT

- ☐ change the width and height of the `TextView` to `wrap-content`
- ☐ Change the text to be more than 1 line of text on the device



user selects **medium** font setting

DESCRIPTION

SAMPLE CODE

A **scale-independent pixel** (sp) is a unit of length for specifying the size of a font of type. Its length depends on the user's preference for font size, set in the Settings app of the Android device.

To respect the user's preferences, you should specify all font sizes in scale-independent pixels. All other measurements should be given in **device-independent pixels** (dp's).

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:padding="8dp"
    android:textSize="20sp"
    android:text="Hello"/>
```

← → ↻ www.google.com/design/spec/style/typography.html#typography-standard-styles ☆ ≡

≡ Style > Typography

Too many type sizes and styles at once can wreck any layout. A typographic scale has a limited set of type sizes that work well together along with the layout grid. The basic set of styles are based on a typographic scale of 12, 14, 16, 20 and 34.

English-like: These sizes and styles are chosen to balance content density and reading comfort under typical usage conditions. Type sizes are specified with sp (scaleable pixels) to enable large type modes for [accessibility](#).

Tall:

- Weight: Use Regular weight, as Medium weight is unavailable in Noto. In addition, Google recommends avoiding Bold weight, based on feedback from native speakers that Bold is too heavy.
- Font size: For Title through Caption styles, font size is 1 px larger than that specified for English. For styles larger than Title, the English type size is suitable.

Dense:

- Weight: Since Noto CJK has seven weights that match Roboto, use the same weight settings as English.
- Font size: For Title through Caption styles, the font size is

Display 4

Display 3

Display 2

Display 1

Headline

Title

Subhead

Body 2

Body 1

Caption

Button

Light 112sp

Regular 56sp

Regular 45sp

Regular 34sp

Regular 24sp

Medium 20sp

Regular 16sp (Device), Regular 15sp (Desktop)

Medium 14sp (Device), Medium 13sp (Desktop)

Regular 14sp (Device), Regular 13sp (Desktop)

Regular 12sp

MEDIUM (ALL CAPS) 14sp

```
android:textAppearance="?android:textAppearanceSmall" />
```

```
android:textAppearance="?android:textAppearanceLarge"
```

Scale Independent Pixels (SP)

- This is a size in Android. It can be used for height, width, text, margins, padding.
- This is the same as DP – Android reads them as equivalent.
- It is intended for font sizes.

4. Circle and correct one problem in each View.

(a) `<TextView`
 `android:layout_width="match_parent"`
 `android:layout_height="wrap_content"`
 `android:text="Motorcycle"`
 `android:padding="25dp"`
 `android:textSize="20sp"`

(b) `<JButton`
 `android:layout_width="wrap_content"`
 `android:layout_height="wrap_content"`
 `android:text="Continue"`
 `android:textSize="40sp" />`

(c) `<TextView`
 `android:layout_width="400dp"`
 `android:layout_height="100dp"`
 `android:text="Driving"`
 `android:textSize="20sp" />;`

- (d)

```
<Image
    android:layout_height="wrap_content"
    android:layout_width="match_parent"
    android:src="@drawable/snowblower" />
```
- (e)

```
<EditText
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:hint="First Name"
    android:layout_height="wrap_content"
    android:layout_gravity="center" />
```
- (f)

```
<Button
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:text="Introduction"
    android:textSize="24sp" />
```