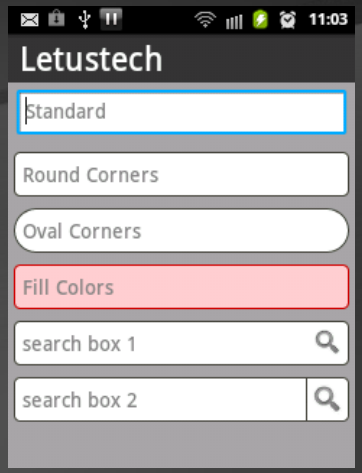
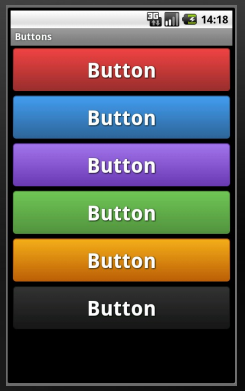
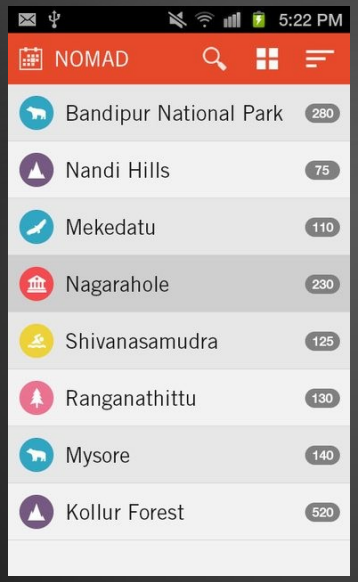
Telerik Academy







Android Views

The building blocks of a framework

Antony Jekov

Telerik Software Academy

http://academy.telerik.com

Table of Contents

1. What is a View and what do we need it for?
2. What is a Canvas?
3. Basic features of a View.
4. Where is the View used?
5. What is a Context and what is it good for?
6. Default Views provided by the Android platform.
7. Extending functionality

View Basic Structure

- The View is the base building block of the Android UI.
- It holds the most basic logic needed for having a visible element on the screen:
  - Position along the x and y axes.
  - Size in width and height.
- It has rectangular form.
- It handles a canvas instance in order to render its contents to the screen.

View Basic Structure

- The view has a layout phase consisting of two steps:
  - measure();
  - layout();
- The final layout decisions are made by the ViewGroup.
- The view can be recognized using an assigned ID.

Views

Live Demo

I can see stuff on my screen.
Where does it come from?

The Canvas class

- The Canvas class is a wrapper around the OpenGL framework that allows as to draw elements and resources on the screen.
- It is responsible for displaying the view elements.
- It is passed by the framework to every view element at the time of “invalidation”.

Canvas

- The canvas draws on a bitmap and then releases this bitmap thus making it visible on the screen.
- Basic draw calls of the canvas:
  - drawLine();
  - drawText();
  - drawRect();
  - drawCircle();
  - drawColor();

Canvas

- Other frequently used methods:
  - save(); saves the current state.
  - restore(); restores previous state.
  - translate(); translates the canvas.
  - clipRect(); clips the canvas.

Telerik Academy

Canvas drawing

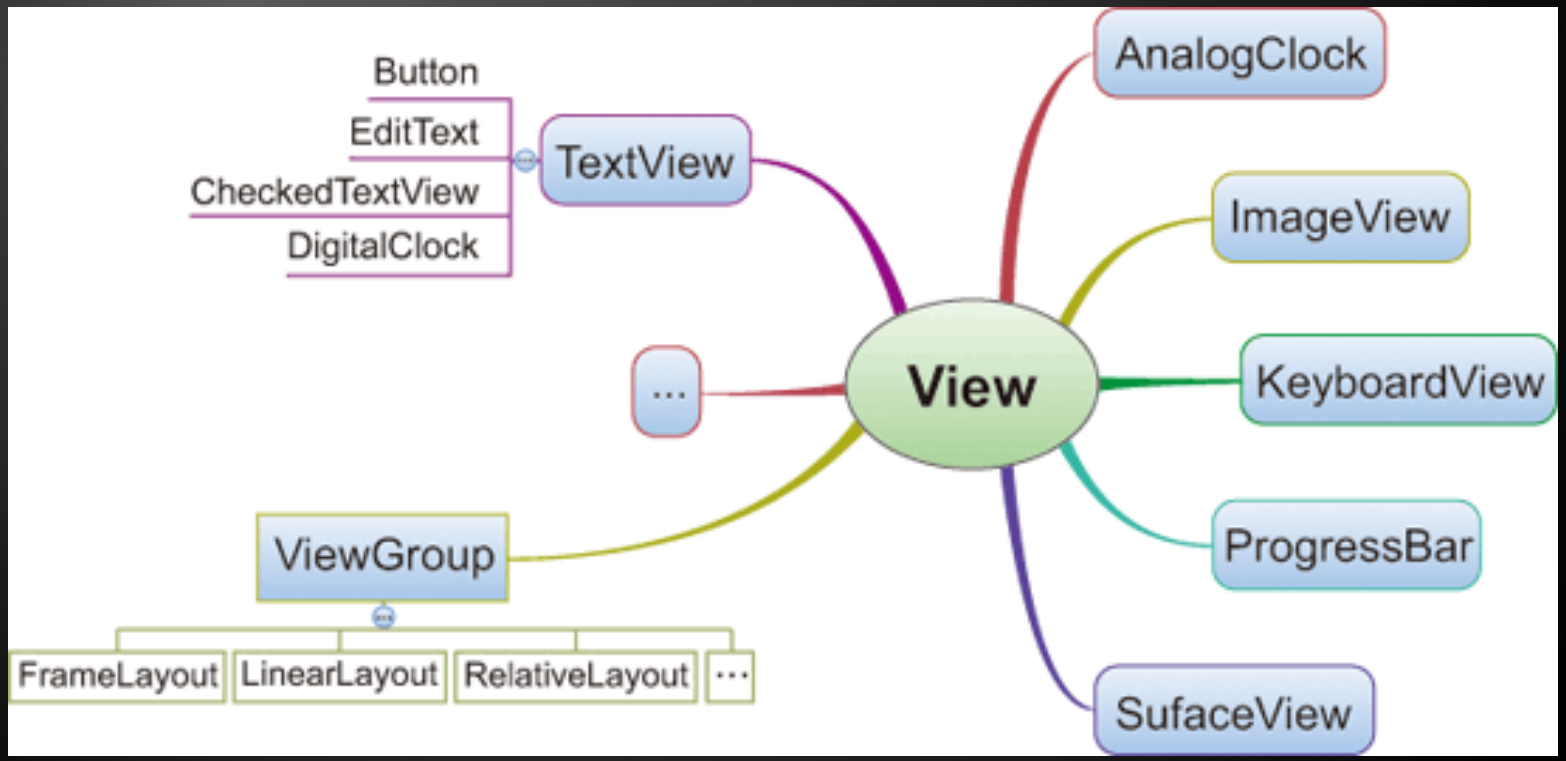
Live Demo

Simple touch handling

- The views are also responsible for handling the user input.
- Gestures are a fundamental part of input in the Android platform.
- MotionEvent class holds the information about a touch event.

```
@Override
public boolean onTouchEvent(MotionEvent
event) {
            return super.onTouchEvent(event);
        }
```

The bigger picture



Telerik Academy

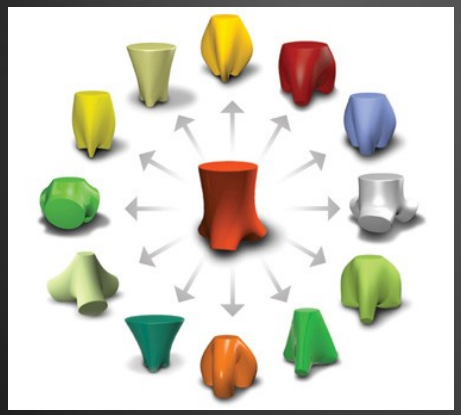
Default Views and basic customizations

Live Demo

Telerik Academy

Still chopsticks, but better...

Extending the default controls and further customizations



Simple touch handling

- All you need to do to extend is... extend.

```
public class MyClock extends AnalogClock {
    @Override
    protected void onLayout(boolean changed,
        int left, int top, int right, int
        bottom) {
        super.onLayout(changed, left, top,
            right, bottom);
    }

    @Override
    protected void onDraw(Canvas canvas) {
        super.onDraw(canvas);
    }
}
```

Simple touch handling

- By overriding the onLayout() it is easy to get the view dimensions when they are being applied.
- Overriding the onDraw() gives access to the canvas and the rendering.
- All that is left – add your brilliant idea to the default view's present logic.

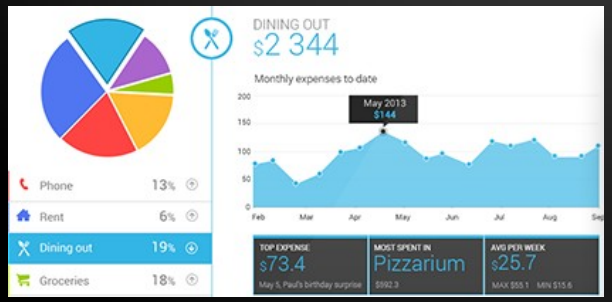
Telerik Academy

Extending the default controls

Live Demo

Telerik Academy

Going rouge!
Extending the view class



Canvas

- Three ways from here
  - Override View (Easy)
    - Good for light tasks like turn based games or some controls.
  - Override SurfaceView (Still Easy)
    - Dedicated for rendering, it could play well for lighter games.
  - Override GlSurfaceView (You die)
    - The heavy stuff, OpenGLES rendering – every good game engine uses it.

Telerik Academy

Custom Views Workshop

Live Demo

Resources

- http://developer.android.com/reference/android/view/View.html
  Views official documentation.
- http://developer.android.com/reference/android/graphics/Paint.html
  Paint documentation.
- http://developer.android.com/reference/android/graphics/Canvas.html
  Canvas documentation.


Questions?

Exercises

1. Extend a default view of your choice. Try to come up with something useful.
2. Draw a nice picture of a house with canvas – Use primitives only, go wild.
3. Create a simple bar chart accepting data, that has category and value and displays this data as vertical bars. The value must be rendered above each bar and the category must be rendered bellow each bar.
4. * Add vertical axis for displaying value steps and a horizontal axis for the categories.