# Measuring Text

27 Mar 2014

If you start manually drawing things to Android's Canvas, you will probably start to draw text as well.

When doing so you need to know where to position the text when you draw, and to do that you will need to measure the text before drawing it, to compute the starting x/y values.

---

In an app recently I needed to draw some text centered both vertically and horizontally on the Canvas. So I started off with the following code:

```
Paint mTextPaint = new Paint();
mTextPaint.setTextAlign(Paint.Align.CENTER); // Center the text

// Later when you draw...
canvas.drawText(mText, // Text to display
        mBounds.centerX(), // Center X of canvas bounds
        mBounds.centerY(), // Center Y of canvas bounds
        mTextPaint
);
```

I didn't expect this to work first-time and it didn't, producing the following:

## Measuring text

Next I tried to position the text, calculating the text's height/width and modifying the drawing X and Y values appropriately:

```
int mTextWidth, mTextHeight; // Our calculated text bounds
Paint mTextPaint = new Paint();

// Now lets calculate the size of the text
Rect textBounds = new Rect();
mTextPaint.getTextBounds(mText, 0, mText.length(), textBounds);
mTextWidth = textBounds.width();
mTextHeight = textBounds.height();

// Later when you draw...
canvas.drawText(mText, // Text to display
        mBounds.centerX() - (mTextWidth / 2f),
        mBounds.centerY() + (mTextHeight / 2f),
        mTextPaint
);
```

This time we're getting much closer, but as you can see that it's not quite centered correctly.

To make sure that I wasn't seeing things, I added an extra call to draw a rectangle behind the text, with the exact bounds calculated with `Paint.getTextBounds()`.

As you can see, the text clearly draws outside of it's calculated bounds, both in height and width.

### Another text measuring method

It was at this point where I saw that Paint has another method for calculating text width: `Paint.measureText()`

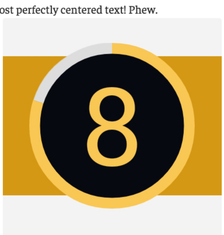
This method only calculates the width and not the height, so I next tried combining the two methods:

```
int mTextWidth, mTextHeight; // Our calculated text bounds
Paint mTextPaint = new Paint();

// Now lets calculate the size of the text
Rect textBounds = new Rect();
mTextPaint.getTextBounds(mText, 0, mText.length(), textBounds);
mTextWidth = mTextPaint.measureText(mText); // Use measureText to calculate w
mTextHeight = textBounds.height(); // Use height from getTextBounds()

// Later when you draw...
canvas.drawText(mText, // Text to display
        mBounds.centerX() - (mTextWidth / 2f),
        mBounds.centerY() + (mTextHeight / 2f),
        mTextPaint
);
```

Which resulted in almost perfectly centered text! Phew.



---

Last updated: 2019-04-10

## Related Posts

Suspending over Views 03 Dec 2019
Suspending over Views—Example 03 Dec 2019
WindowInsets — Listeners to layouts 12 Apr 2019