

# HenCoder Plus 第 7 课 讲义

---

## 绘制（二）

---

### 文字的测量

- 绘制文字：drawText()

#### 文字测量难点之一：居中的纵向测量

- 方式一：Paint.getTextBounds() 之后，使用  $(\text{bounds.top} + \text{bounds.bottom}) / 2$
- 方式二：Paint.getFontMetrics() 之后，使用  $(\text{fontMetrics.ascend} + \text{fontMetrics.descend}) / 2$



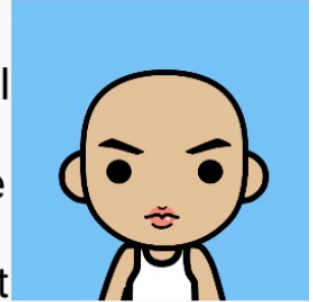
#### 文字测量难点之二：左对齐

- 用 `getTextBounds()` 之后的 `left` 来计算

### 文字测量难点之三：换行

- 用 `breakText()` 来计算

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean justo sem, sollicitudin in maximus a, vulputate id magna. Nulla non quam a massa sollicitudin commodo fermentum et est. Suspendisse potenti. Praesent dolor dui, dignissim quis tellus tincidunt, porttitor vulputate nisl. Aenean tempus lobortis finibus. Quisque nec nisl laoreet, placerat metus sit amet, consectetur est. Donec nec quam tortor. Aenean aliquet dui in enim venenatis, sed luctus ipsum maximus. Nam feugiat nisi rhoncus lacus facilisis pellentesque nec vitae lorem. Donec et risus eu ligula dapibus lobortis vel vulputate turpis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; In porttitor, risus aliquam rutrum finibus, ex mi ultricies arcu, quis ornare lectus tortor nec metus. Donec ultricies metus at magna cursus congue. Nam eu sem eget enim pretium venenatis. Duis nibh ligula, lacinia ac nisi vestibulum, vulputate lacinia tortor.



## Canvas 的范围裁切

- `clipRect()`
- `clipPath()` `clipPath()` 切出来的圆为什么没有抗锯齿效果？因为「强行切边」
- `clipOutRect()` / `clipOutPath()`

## Canvas 的几何变换

- `translate(x, y)`
- `rotate(degree)`
- `scale(x, y)`
- `skew(x, y)`

重点：Canvas 的几何变换方法参照的是 View 的坐标系，而绘制方法（`drawXxx()`）参照的是 Canvas 自己的坐标系。

### 关于多重变换：

Canvas 的变换方法多次调用的时候，由于 Canvas 的坐标系会整体被变换，因此当平移、旋转、缩放、错切等变换多重存在的时候，Canvas 的变换参数会非常难以计算，因此可以改用倒序的理解方式：

将 Canvas 的变换理解为 Canvas 的坐标系不变，每次变换是只对内部的绘制内容进行变换，同时把 Canvas 的变换顺序看作是倒序的（即写在下面的变换先执行），可以更加方便进行多重变换的参数计算。

## Matrix 的几何变换

- `preTranslate(x, y)` / `postTranslate(x, y)`
- `preRotate(degree)` / `postRotate(degree)`
- `preScale(x, y)` / `postScale(x, y)`
- `preSkew(x, y)` / `postSkew(x, y)`

其中 `preXxx()` 效果和 Canvas 的准同名方法相同，`postXxx()` 效果和 Canvas 的准同名方法顺序相反。

### 注意

如果多次重复使用 Matrix，在使用之前需要用 `Matrix.reset()` 来把 Matrix 重置。

## 使用 Camera 做三维旋转

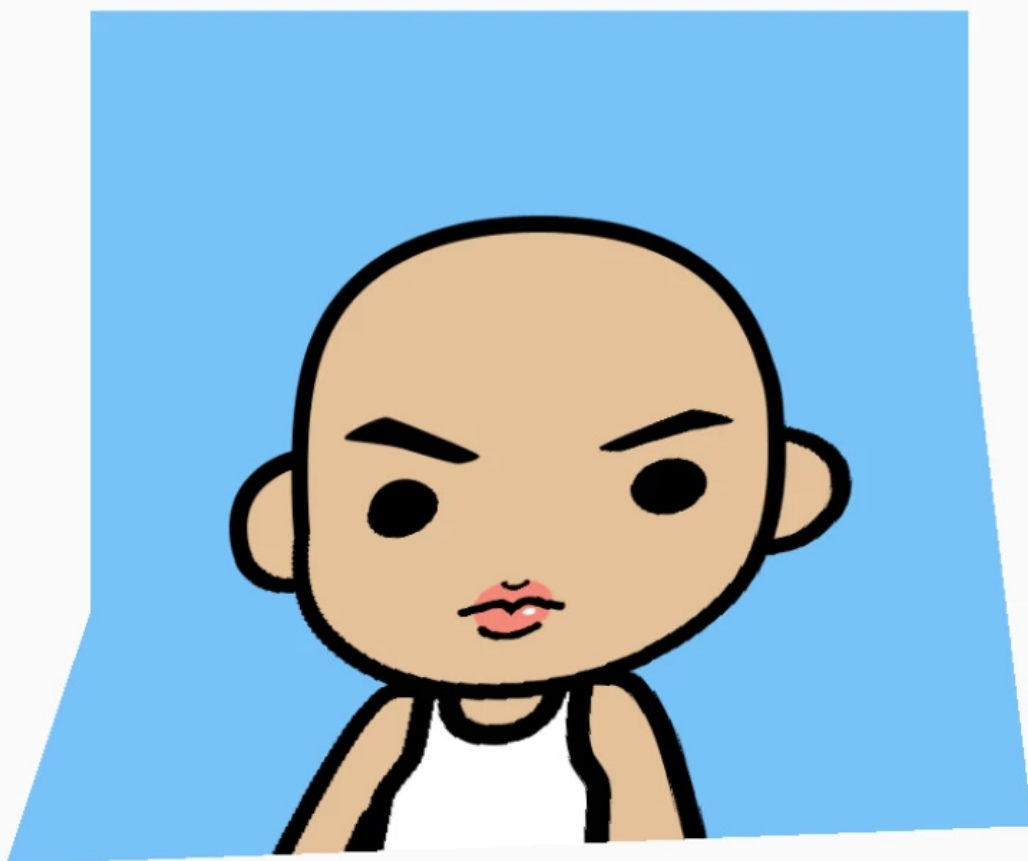
- `rotate()` / `rotateX()` / `rotateY()` / `rotateZ()`
- `translate()`
- `setLocation()`

其中，一般只用 `rotateX()` 和 `rotateY()` 来做沿 x 轴或 y 轴的旋转，以及使用 `setLocation()` 来调整放缩的视觉幅度。

对 Camera 变换之后，要用 `Camera.applyToCanvas(Canvas)` 来应用到 Canvas。

### `setLocation()`

这个方法一般前两个参数都填 0，第三个参数为负值。由于这个值的单位是硬编码写死的，因此像素密度越高的手机，相当于 Camera 距离 View 越近，所以最好把这个值写成与机器的 density 成正比的一个负值，例如 `-6 * density`。



## 问题和建议？

课上技术相关的问题，都可以在学员群里和大家讨论，我一旦有时间也都会来解答。如果我没来就 @我一下吧！

具体技术之外的问题和建议，都可以找丢物线（微信：diuwuxian），丢丢会为你解答技术以外的一切。



## 更多内容:

- 网站: <https://hencoder.com>
- 微信公众号: HenCoder

# HenCoder

## 给高级 Android 工程师的进阶手册

微信公众号: HenCoder

微博：扔物线

知乎专栏: HenCoder

## 稀土掘金：扔物线

<http://hencoder.com>

