PathMeasure

顾名思义，PathMeasure是一个用来测量Path的类，主要有以下方法:

构造方法

| **方法名** | **释义** |
| --- | --- |
| PathMeasure() | 创建一个空的PathMeasure |
| PathMeasure(Path path, boolean forceClosed) | 创建 PathMeasure 并关联一个指定的Path(Path需要已经创建完成)。 |

公共方法

| **返回值** | **方法名** | **释义** |
| --- | --- | --- |
| void | setPath(Path path, boolean forceClosed) | 关联一个Path |
| boolean | isClosed() | 是否闭合 |
| float | getLength() | 获取Path的长度 |
| boolean | nextContour() | 跳转到下一个轮廓 |
| boolean | getSegment(float startD, float stopD, Path dst, boolean startWithMoveTo) | 截取片段 |
| boolean | getPosTan(float distance, float[] pos, float[] tan) | 获取指定长度的位置坐标及该点  切线值tangle |
| boolean | getMatrix(float distance, Matrix matrix, int flags) | 获取指定长度的位置坐标及该点Matrix(矩阵) |

PathMeasure的方法也不多，接下来我们就逐一的讲解一下。

1.构造函数

构造函数有两个。

**无参构造函数：**

PathMeasure ()

* 1
* 1

用这个构造函数可创建一个空的 PathMeasure，但是使用之前需要先调用 setPath 方法来与 Path 进行关联。被关联的 Path 必须是已经创建好的，如果关联之后 Path 内容进行了更改，则需要使用 setPath 方法重新关联。

**有参构造函数：**

PathMeasure (Path path, boolean forceClosed)

* 1
* 1

用这个构造函数是创建一个 PathMeasure 并关联一个 Path， 其实和创建一个空的 PathMeasure 后调用 setPath 进行关联效果是一样的，同样，被关联的 Path 也必须是已经创建好的，如果关联之后 Path 内容进行了更改，则需要使用 setPath 方法重新关联。

该方法有两个参数，第一个参数自然就是被关联的 Path 了，第二个参数是用来确保 Path 闭合，如果设置为 true， 则不论之前Path是否闭合，都会自动闭合该 Path(如果Path可以闭合的话)。

**在这里有两点需要明确:**

* 1. 不论 forceClosed 设置为何种状态(true 或者 false)， 都不会影响原有Path的状态，**即 Path 与 PathMeasure 关联之后，之前的的 Path 不会有任何改变。**
  2. forceClosed 的设置状态可能会影响测量结果，**如果 Path 未闭合但在与 PathMeasure 关联的时候设置 forceClosed 为 true 时，测量结果可能会比 Path 实际长度稍长一点，获取到到是该 Path 闭合时的状态。**

2.setPath、 isClosed 和 getLength

这三个方法都如字面意思一样，非常简单，这里就简单是叙述一下，不再过多讲解。

setPath 是 PathMeasure 与 Path 关联的重要方法，效果和 构造函数 中两个参数的作用是一样的。

isClosed 用于判断 Path 是否闭合，但是如果你在关联 Path 的时候设置 forceClosed 为 true 的话，这个方法的返回值则一定为true。

getLength 用于获取 Path 的总长度，在之前的测试中已经用过了。

3.getSegment

getSegment 用于获取Path的一个片段，方法如下：

boolean getSegment (float startD, float stopD, Path dst, boolean startWithMoveTo)

* 1
* 1

方法各个参数释义：

| **参数** | **作用** | **备注** |
| --- | --- | --- |
| 返回值(boolean) | 判断截取是否成功 | true 表示截取成功，结果存入dst中，false 截取失败，不会改变dst中内容 |
| startD | 开始截取位置距离 Path 起点的长度 | 取值范围: 0 <= startD < stopD <= Path总长度 |
| stopD | 结束截取位置距离 Path 起点的长度 | 取值范围: 0 <= startD < stopD <= Path总长度 |
| dst | 截取的 Path 将会添加到 dst 中 | 注意: 是添加，而不是替换 |
| startWithMoveTo | 起始点是否使用 moveTo | 用于保证截取的 Path 第一个点位置不变 |

* 如果 startD、stopD 的数值不在取值范围 [0, getLength] 内，或者 startD == stopD 则返回值为 false，不会改变 dst 内容。
* 如果在安卓4.4或者之前的版本，在默认开启硬件加速的情况下，更改 dst 的内容后可能绘制会出现问题，请关闭硬件加速或者给 dst 添加一个单个操作，例如: dst.rLineTo(0, 0)
* 可以用以下规则来判断 startWithMoveTo 的取值：

| **取值** | **主要功用** |
| --- | --- |
| true | 保证截取得到的 Path 片段不会发生形变 |
| false | 保证存储截取片段的 Path(dst) 的连续性 |

### 4.nextContour

我们知道 Path 可以由多条曲线构成，但不论是 getLength , getgetSegment 或者是其它方法，都只会在其中第一条线段上运行，而这个 nextContour 就是用于跳转到下一条曲线到方法，如果跳转成功，则返回 true， 如果跳转失败，则返回 false。

#### 5.getPosTan

这个方法是用于得到路径上某一长度的位置以及该位置的正切值：

boolean getPosTan (float distance, float[] pos, float[] tan)

* 1
* 1

方法各个参数释义：

| **参数** | **作用** | **备注** |
| --- | --- | --- |
| 返回值(boolean) | 判断获取是否成功 | true表示成功，数据会存入 pos 和 tan 中， false 表示失败，pos 和 tan 不会改变 |
| distance | 距离 Path 起点的长度 | 取值范围: 0 <= distance <= getLength |
| pos | 该点的坐标值 | 坐标值: (x==[0], y==[1]) |
| tan | 该点的正切值 | 正切值: (x==[0], y==[1]) |

* 1.**通过 tan 得值计算出图片旋转的角度**，tan 是 tangent 的缩写，即中学中常见的正切， 其中tan[0](http://blog.csdn.net/u013831257/article/details/x)是邻边边长，tan[1](http://blog.csdn.net/u013831257/article/details/y)是对边边长，而Math中 atan2 方法是根据正切是数值计算出该角度的大小,得到的单位是弧度，所以上面又将弧度转为了角度。

6.getMatrix

这个方法是用于得到路径上某一长度的位置以及该位置的正切值的矩阵：

boolean getMatrix (float distance, Matrix matrix, int flags)

* 1
* 1

方法各个参数释义：

| **参数** | **作用** | **备注** |
| --- | --- | --- |
| 返回值(boolean) | 判断获取是否成功 | true表示成功，数据会存入matrix中，false 失败，matrix内容不会改变 |
| distance | 距离 Path 起点的长度 | 取值范围: 0 <= distance <= getLength |
| matrix | 根据 falgs 封装好的matrix | 会根据 flags 的设置而存入不同的内容 |
| flags | 规定哪些内容会存入到matrix中 | 可选择 POSITION\_MATRIX\_FLAG(位置)  ANGENT\_MATRIX\_FLAG(正切) |