# TextView富文本

原创 xyzso1z 2020-07-19 01:56:26 15 收藏 原力计划 编辑 版权

分类专栏： Android 文章标签： android TextView 富文本 SpannableString Html.fromHtml

## 一、富文本介绍

`TextView` 富文本显示主要有两种方式：

1. `SpannableString` 类
2. `html`

下面介绍这两种方式实现

## 二、`SpannableString` 类

### 2.1 作用：

1. 修改字体( `StyleSpan` )[粗体、斜体等]
2. 文本字体( `TypefaceSpan` )
3. 修改文字颜色( `ForegroundColorSpan` )
4. 文字绝对大小( `AbsoluteSizeSpan` )
5. 文字相对大小( `RelativeSizeSpan` )
6. 图片( `ImageSpan` )
7. 文本可点击( `ClickableSpan` )
8. 文本超链接( `URLSpan` )
9. 背景色 ( `BackgroundColorSpan` )
10. 下划线( `UnderlineSpan` )
11. 中划线( `StrikethroughSpan` )
12. 光栅效果( `RasterizerSpan` )
13. 修饰效果( `MaskFilterSpan` )
14. 下标( `SubscriptSpan` )
15. 上标( `SuperscriptSpan` )
16. 基于x轴缩放( `ScaleXSpan` )
17. 文本外貌( `TextAppearanceSpan` )

### 2.2 `setSpan()`

```
void setSpan (Object what, int start, int end, int flags)
```

- 参数说明

| 参数 | 类型 | 说明 |
| --- | --- | --- |
| `what` | `Object` | 样式 |
| `start` | `int` | 样式开始的字符索引 |
| `end` | `int` | 样式结束的字符索引 |
| `flags` | `int` | 新插入字符的设置 |

- `flags` 值

| 取值 | 说明 |
| --- | --- |
| `Spanned.SPAN_EXCLUSIVE_EXCLUSIVE` | 前后都不 |

| 取值 | 说明 |
| --- | --- |
| `Spanned.SPAN_EXCLUSIVE_INCLUSIVE` | 前面不包括， |
| `Spanned.SPAN_INCLUSIVE_EXCLUSIVE` | 前面包括，后 |
| `Spanned.SPAN_INCLUSIVE_INCLUSIVE` | 前后都包 |

举例说明各参数：

```
1
```

### 2.3 修改字体（`StyleSpan`）

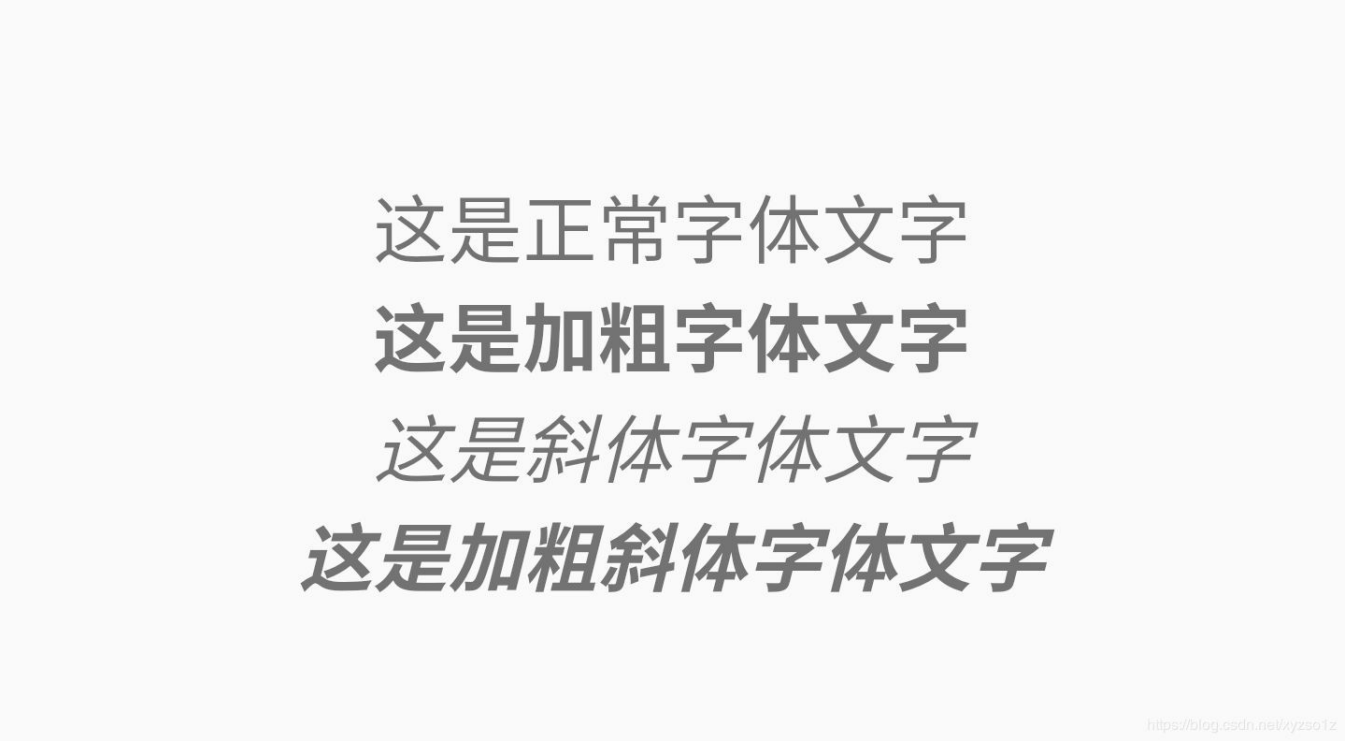
- 作用：修改文字字体(粗体、斜体、粗斜体)
- 使用：

```
1 // 把字体改为粗体 Typeface.BOLD
2 textSpanned.setSpan(new StyleSpan(Typeface.BOLD), 0, 10, Spanned.SPAN_EXCLUSIVE_INCLUSIVE);
```

- 参数说明 `Typeface.*`

| 参数 | |
| --- | --- |
| `Typeface.NORMAL` | |
| `Typeface.BOLD` | |
| `Typeface.ITALIC` | |
| `Typeface.BOLD_ITALIC` | |

- 效果：




### 2.4 文本字体（`TypefaceSpan`）

- 作用： 修改文字字体格式(不同于`StyleSpan`)，
- 使用：
  **使用系统自带**
  系统自带`monospace`、`serif`、`sans-serif`

```
SpannableString sansSerifText = new SpannableString("这是sans-serif字体\n");
TypefaceSpan sansSerif = new TypefaceSpan("sans-serif");
sansSerifText.setSpan(sansSerif, 0, sansSerifText.length(), Spanned.SPAN_INCLUSIVE_EXCLUSIVE);
```

**使用自定义字体**

1. 把字体文件放入`src\main\assets\`下
2. 加载`Typeface`

```
public class TypefaceUtil {

private static final HashMap<String, Typeface> sCachedFonts = new  HashMap<String, Typeface>();
private static final String PREFIX_ASSET = "asset:";

private TypefaceUtil() {
}

/**
 * @param familyName if start with 'asset:' prefix, then load font from asset folder.
 */
public static Typeface load(Context context, String familyName, int style) {
    if (familyName != null && familyName.startsWith(PREFIX_ASSET)) {
        synchronized (sCachedFonts) {
            try {
                if (!sCachedFonts.containsKey(familyName)) {
                    final Typeface typeface = Typeface.createFromAsset(context.getAssets(), familyName.substring(PREFIX
                    sCachedFonts.put(familyName, typeface);
                    return typeface;
                }
            } catch (Exception e) {
                e.printStackTrace();
                return Typeface.DEFAULT;
            }
            return sCachedFonts.get(familyName);
        }
    }
    return Typeface.create(familyName, style);
  }
}
```

3. 实现`MetricAffectingSpan`类

```
static class CustomTypefaceSpan extends MetricAffectingSpan {

  private final Typeface typeface;

  CustomTypefaceSpan(final Typeface typeface) {
      this.typeface = typeface;
  }

  @Override
  public void updateDrawState(final TextPaint drawState) {
      apply(drawState);
  }

  @Override
  public void updateMeasureState(final TextPaint paint) {
      apply(paint);
  }

  private void apply(final Paint paint) {
      final Typeface oldTypeface = paint.getTypeface();
      final int oldStyle = oldTypeface != null ? oldTypeface.getStyle() : 0;
      final int fakeStyle = oldStyle & ~typeface.getStyle();

      if ((fakeStyle & Typeface.BOLD) != 0) {
          paint.setFakeBoldText(true);
```

```
        }

        if ((fakeStyle & Typeface.ITALIC) != 0) {
            paint.setTextSkewX(-0.25f);
        }

        paint.setTypeface(typeface);
    }
}
```

4. 设置 `SpannableString`

```
SpannableString customText0 = new SpannableString("这是自定义字体1\n");
CustomTypefaceSpan customTypefaceSpan0 = new CustomTypefaceSpan(TypefaceUtil.load(this, "asset:text.ttf", 0));
customText0.setSpan(customTypefaceSpan0, 0, customText0.length(), Spannable.SPAN_EXCLUSIVE_EXCLUSIVE);
```

- 效果


## 2.5 文本颜色（`ForegroundColorSpan`）

- 作用：修改文字颜色
- 使用：

```
SpannableString spannableString = new SpannableString("设置文本颜色");
spannableString.setSpan(new ForegroundColorSpan(Color.BLUE), 0, spannableString .length(), Spanned.SPAN_EXCLUSIVE_IN
textView.setText(spannableString );
```

- 效果：


## 2.6 设置文字大小（`AbsoluteSizeSpan`、`RelativeSizeSpan`）

- 作用：修改文字尺寸
- 使用：

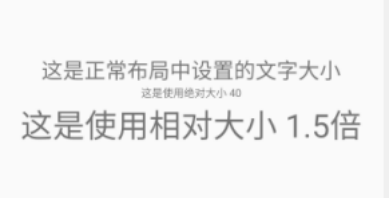
1. 绝对大小 `AbsoluteSizeSpan`

```
SpannableString spannableString = new SpannableString("使用绝对大小设置文字尺寸");
spannableString.setSpan(new AbsoluteSizeSpan(40), 0,2, Spanned.SPAN_INCLUSIVE_INCLUSIVE);
```

2. 相对大小

```
SpannableString spannableString = new SpannableString("使用相对大小设置文字尺寸");
spannableString.setSpan(new RelativeSizeSpan(1.5F), 0, 2, Spanned.SPAN_INCLUSIVE_INCLUSIVE);
```

- 效果：




### 2.7 设置图片（`ImageSpan`）

- 作用：在文字中插入图片
- 使用：

1. 图片底部与文字底部对齐

```
    Drawable drawable = getDrawable(R.mipmap.icon);
    drawable.setBounds(0, 0, drawable.getMinimumWidth(), drawable.getMinimumHeight());

    ImageSpan imageSpan = new ImageSpan(drawable);
    sp.setSpan(imageSpan, 0, 1, ImageSpan.ALIGN_BASELINE);
```

2. 自定义 使图片中部与文字对齐

```
public class CenterAlignImageSpan extends ImageSpan {
  public CenterAlignImageSpan(Drawable drawable) {
    super(drawable);
  }

  @Override
  public void draw(Canvas canvas, CharSequence text, int start, int end, float x, int top, int y, int bottom, Paint pain
    Drawable b = getDrawable();
    Paint.FontMetricsInt fm = paint.getFontMetricsInt();
    //计算y方向的位移
    int transY = (y + fm.descent + y + fm.ascent) / 2 - b.getBounds().bottom / 2;
    canvas.save();
    //绘制图片位移一段距离
    canvas.translate(x, transY);
    b.draw(canvas);
    canvas.restore();
  }
}

//使用
CenterAlignImageSpan imageSpan1 = new CenterAlignImageSpan(drawable);
sp.setSpan(imageSpan1,0, 1, ImageSpan.ALIGN_CENTER);
```

- 效果

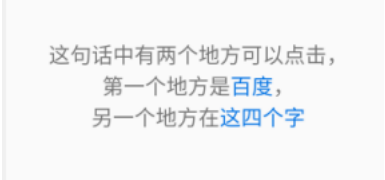

### 2.8 设置文本可点击（`ClickableSpan`）

- 作用：设置文字可以接收点击事件
- 使用：

```
    textView.setTextIsSelectable(true);
    //方法重新设置文字点击
    textView.setMovementMethod(LinkMovementMethod.getInstance());
    //方法重新设置文字背景为透明色。
    textView.setHighlightColor(getResources().getColor(android.R.color.darker_gray));

    String text = "这句话中有两个地方可以点击，\n第一个地方是百度，\n另一个地方在这四个字";
```

```
        SpannableString spannableString = new SpannableString(text);
        spannableString.setSpan(new Clickable(firstClickListener), text.indexOf("百度"), text.indexOf(", \n另"), Spanned.SPAN
        spannableString.setSpan(new Clickable(secondClickListener), text.indexOf("这四个字"), text.length(), Spanned.SPAN_EX

    private View.OnClickListener firstClickListener = new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Uri uri = Uri.parse(url);
            Context context = v.getContext();
            Intent intent = new Intent(Intent.ACTION_VIEW, uri);
            intent.putExtra(Browser.EXTRA_APPLICATION_ID, context.getPackageName());
            context.startActivity(intent);
        }
    };

    private View.OnClickListener secondClickListener = new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Toast.makeText(ClickableActivity.this, "第二个点击事件", Toast.LENGTH_SHORT).show();

        }
    };

    static class Clickable extends ClickableSpan {
        View.OnClickListener listener;

        Clickable(View.OnClickListener listener) {
            this.listener = listener;
        }

        @Override
        public void onClick(@NonNull View v) {
            listener.onClick(v);
        }

        @Override
        public void updateDrawState(@NonNull TextPaint ds) {
            super.updateDrawState(ds);
            ds.setColor(0xFF0474E5);//设置文字颜色
            ds.setUnderlineText(false);    //去除超链接的下划线
            ds.clearShadowLayer();
        }
    }

```

- 效果




### 2.9 文本超链接（URLSpan）

- 作用：设置点击文字打开相应链接
- 使用：

```
        textView.setTextIsSelectable(true);
        //方法重新设置文字点击
        textView.setMovementMethod(LinkMovementMethod.getInstance());
        //方法重新设置文字背景为透明色。
        textView.setHighlightColor(getResources().getColor(android.R.color.darker_gray));

        SpannableString ss = new SpannableString("百度一下");
        ss.setSpan(new URLSpan("http://www.baidu.com"), 0, 2, Spanned.SPAN_EXCLUSIVE_EXCLUSIVE);
```

- 效果：


## 2.10 背景色（`BackgroundColorSpan`）

- 作用：修改文字背景色
- 使用

```
String text = "这个字背景是蓝色的,\n这个字背景是红色的";
SpannableString spannableString = new SpannableString(text);
spannableString.setSpan(new BackgroundColorSpan(0xff00ddff), 6, 8, Spannable.SPAN_INCLUSIVE_INCLUSIVE);
spannableString.setSpan(new BackgroundColorSpan(0xffcc0000), text.length() - 3, text.length() - 1, Spannable.SPAN_IN

textView.setText(spannableString);
```

- 效果


## 2.11 下划线（`UnderlineSpan`）

- 作用：在文字下方显示下划线
- 使用：

```
String text = "下面这句话是重点，\n记得划线";
SpannableString spannableString = new SpannableString(text);
spannableString.setSpan(new UnderlineSpan(), text.length() - 4, text.length(), Spannable.SPAN_EXCLUSIVE_EXCLUSIVE);
textView.setText(spannableString );
```

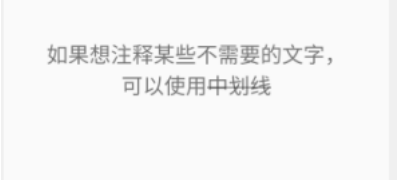
- 效果


## 2.12 中划线（`StrikethroughSpan`）

- 作用：在文字中间加中划线
- 使用：

```
String text = "如果想注释某些不需要的文字，\n可以使用中划线";
SpannableString spannableString = new SpannableString(text);
spannableString.setSpan(new StrikethroughSpan(), text.length() - 3, text.length(), Spannable.SPAN_EXCLUSIVE_EXCLUSIVE
textView.setText(spannableString );
```

- 效果：




### 2.13 修饰效果（`MaskFilterSpan`）

- 作用：给字体加阴影
- 参数：NORMAL、SOLID、OUTER、INNER
- 使用：

```
SpannableString spannableString = new SpannableString("修饰效果 NORMAL");
spannableString.setSpan(new MaskFilterSpan(new BlurMaskFilter(4, BlurMaskFilter.Blur.SOLID)), 0, 4,Spannable.SPAN_EXCL
textView.setText(spannableString );
```

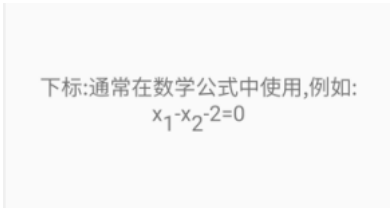
- 效果：


### 2.14 下标（`SubscriptSpan`）

- 作用：让文字处于下部
- 使用：

```
String text = "下标:通常在数学公式中使用,例如:\nx1-x2-2=0";
SpannableString spannableString = new SpannableString(text);
spannableString.setSpan(new SubscriptSpan(), text.indexOf("1"), text.indexOf("1") + 1, Spannable.SPAN_EXCLUSIVE_EXCLU
spannableString.setSpan(new SubscriptSpan(), text.indexOf("2"), text.indexOf("2") + 1, Spannable.SPAN_EXCLUSIVE_EXCLU
textView.setText(spannableString );
```

- 效果：

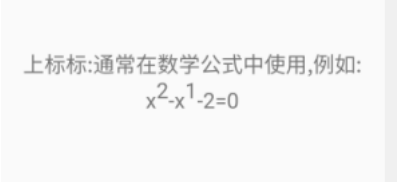



### 2.15 上标（`SuperscriptSpan`）

- 作用：让文字处于上部
- 使用：

```
String text = "上标标:通常在数学公式中使用,例如:\nx2-x1-2=0";
SpannableString spannableString = new SpannableString(text);
spannableString.setSpan(new SuperscriptSpan(), text.indexOf("1"), text.indexOf("1") + 1, Spannable.SPAN_EXCLUSIVE_EX
spannableString.setSpan(new SuperscriptSpan(), text.indexOf("2"), text.indexOf("2") + 1, Spannable.SPAN_EXCLUSIVE_EX
textView.setText(spannableString );
```

- 效果：




### 2.16 基于x轴缩放（`ScaleXSpan`）

- 作用：让文字在x轴方向缩放
- 使用：

```
String text = "基于x轴进行缩放";
SpannableString spannableString = new SpannableString(text);
spannableString.setSpan(new ScaleXSpan(1.6F), text.length() - 2, text.length(), Spannable.SPAN_EXCLUSIVE_EXCLUSIVE);
textView.setText(spannableString );
```

- 效果：


### 2.17 文本样式（`TextAppearanceSpan`）

- 作用：设置文字style
- 使用：

1. style

```
<resources>
  <style name="style_black">
     <item name="android:textSize">30sp</item>
     <item name="android:textColor">@android:color/black</item>
     <item name="android:textStyle">bold</item>
   </style>
</resources>
```

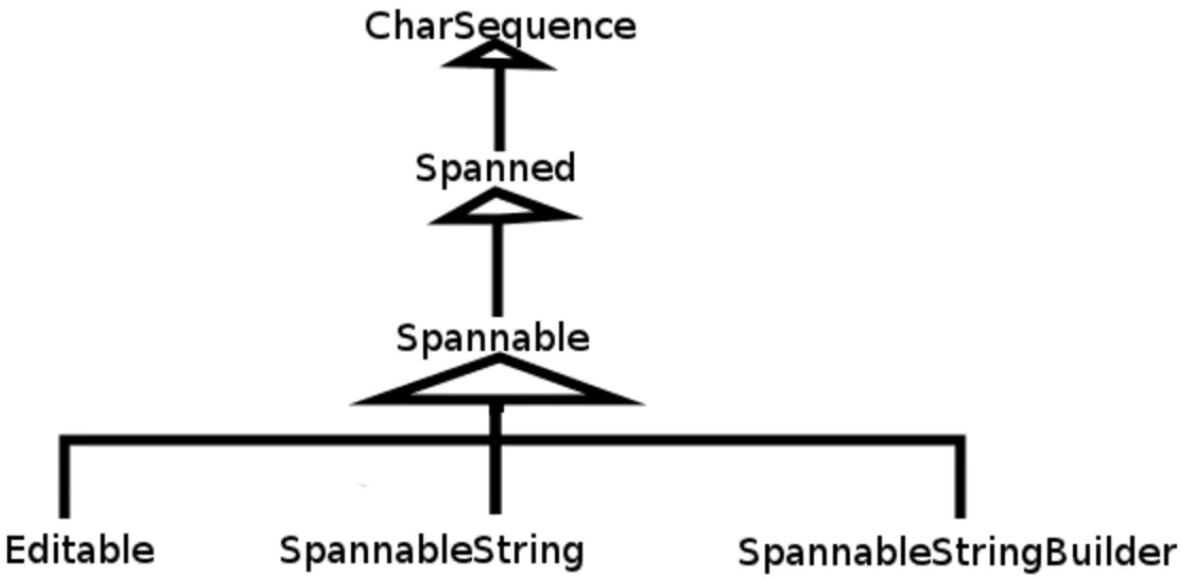
2. 设置

```
String text = "文本外貌是什么样式";
SpannableString spannableString = new SpannableString(text);
spannableString.setSpan(new TextAppearanceSpan(this, R.style.style_black), text.length() - 2, text.length(), Spannabl
textView.setText(spannableString );
```

- 效果：


## 三、`SpannableStringBuilder` 类

- 作用：可以进行拼接`String`、`SpannableString`等，方便操作，提升性能。
- 类图：




- 使用举例：

```
SpannableStringBuilder spannableStringBuilder = new SpannableStringBuilder();

//拼接String 类型
String string = "这是一个String字符串\n";
spannableStringBuilder.append(string);

//拼接SpannableString
SpannableString spannableString = new SpannableString("这是要给SpannableString字符串");
spannableString.setSpan(new ForegroundColorSpan(0x80323232), 0, spannableString.length(), Spanned.SPAN_EXCLUSIVE_IN
spannableStringBuilder.append(spannableString);

textView.setText(spannableStringBuilder);
```

## 四、Html

- 简介：在html中设置文字样式是非常方便的，在Android中有 `Html.fromHtml(String)` 方法可以把html格式解析转换成Spanned就可以使TextView解析显示不同的样式。
- 举例：

```
String string = "这是正常文字<p><font color='red' size='20'> html 设置文字颜色、大小</font>";
textView.setText(Html.fromHtml(string));
```

- 效果


## 五、资源

- 本文源码
- Android知识图谱总结