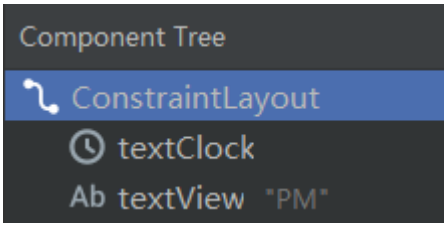
# 第一次作业：

## Timer + TimerTask 实现数字时钟

1.运行效果展示


2.布局结构

布局主要使用TextClock组件显示主体,TextView组件显示AM/PM信息。



### 3.思路分析与代码展示

1）实现全屏效果并修改 ActionBar 标题信息

```
getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
        WindowManager.LayoutParams.FLAG_FULLSCREEN); //全屏显示
ActionBar actionBar = getSupportActionBar();
actionBar.setTitle("TimerDemo01");
```

2）介绍 TextClock

TextClock 组件可以将格式化模式字符串对应显示为当前日期和时间并更新

3）构造格式化模式字符串

HH 显示小时；mm 显示分钟；ss 显示秒；EE 显示星期；yyyy/MM/dd 显示 年/月/日。

使用 SpannableString 在字符序列基础上对指定的字符进行润饰来替代使用多 TextView 改变文字样式

```
String clockFormatter = "  HH : mm ss\n      EE            yyyy/MM/dd";
Spannable s = new SpannableString(clockFormatter);
s.setSpan(new RelativeSizeSpan( proportion: 0.5f), start: 10, end: 11,
        Spannable.SPAN_INCLUSIVE_INCLUSIVE); //将ss(对应秒)缩小显示
s.setSpan(new RelativeSizeSpan( proportion: 0.3f), start: 12, end: 43,
        Spannable.SPAN_INCLUSIVE_INCLUSIVE); //将星期和日期缩小显示
```

4）使用 TextClock

设置时区，并传入格式化字符串

```
textClock.setTimeZone("GMT+8"); //设置时区
textClock.setFormat24Hour(s);
```

5）使用 Timer+TimerTask 定时更新 AM/PM 信息

在 TimerTask 线程调用 runOnUiThread 更新 UI，获取 textClock 的小时信息并判断修改 textView 组件的内容，实现 AM/PM 的切换，最后使用 Timer 的 schedule 方法在指定定时周期调度 TimerTask

```
timer = new Timer();
timerTask = new TimerTask() {
    @Override
    public void run() {
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                String hourStr = textClock.getText().toString().split( regex: ":")[0].trim();
                int hour = Integer.parseInt(hourStr);
                if (hour >= 0 && hour < 12) {
                    textView.setText("AM");
                } else textView.setText("PM");
            }
        });
    }
};
timer.schedule(timerTask,  delay: 0,  period: 600);
```

## 4.实验总结

通过本次实验，我掌握了如何使用 Timer+TimerTask 实现定时任务，并且如果在其他线程中要更新 UI 时可以使用 onUiThread 实现；学习了 TextClock 组件的使用方法，同时学习了 SpannableString 在字符层面对样式的修改，适用于一段文字需要设置非常多样式的情况。