Android 原生开发大作业——Time 时钟

目录

一、 任务分析

1.1 功能需求

-实时显示当前时间。
-在整点时刻自动更换背景图片。
-支持自定义字体，使用电子数字字体显示时间。
-界面设计需符合用户体验原则，确保操作简便、视觉舒适。

1.2 开发环境

编程语言: Java
开发工具: Android Studio
依赖库： Java21 标准库、android API 35

1.3 预期成果

一个功能完善的数字时钟应用程序，具备实时更新时间、整点更换背景图片、使用电子数字字体显示时间等功能。
用户界面美观，操作流畅，用户体验良好。

1.4 挑战与解决方案

-挑战：如何在整点时刻准确更换背景图片。
解决方案：通过 TimerTask 定时检查当前时间，当分钟数为 0 时触发更换背景图片的操作。
-挑战：如何确保时间显示的准确性和实时性。
解决方案：使用 Calendar 类获取当前时间，并通过 Timer 每秒更新一次显示。

二、 界面实现

首先我在 `/res/ drawable` 中保存了来自网络上的 6 张照片，并存在一个 int 数组内：

```
// 配置每个时间段对应的背景图片
private final int[] hourBackgrounds = {
        R.drawable.img10_14, R.drawable.img6_10, R.drawable.img10_14,
R.drawable.img14_18,
        R.drawable.img18_22, R.drawable.img22_2,
};
```

然后通过当前的小时值来对应 6 张图片，即每四小时切换一次，对应代码如下：

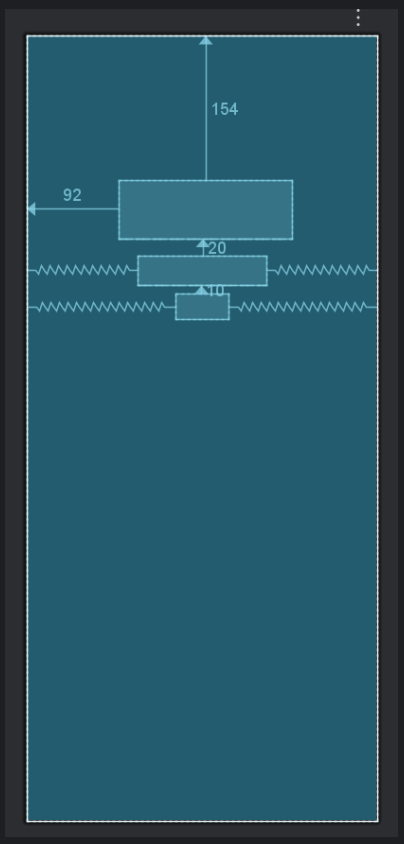
```
private void updateBackground(int hour) {
    // 根据当前小时选择合适的背景
    int backgroundIndex;

    if (hour >= 2 && hour < 6) {
        backgroundIndex = 0;
    } else if (hour >= 6 && hour < 10) {
        backgroundIndex = 1;
    } else if (hour >= 10 && hour < 14) {
        backgroundIndex = 2;
    } else if (hour >= 14 && hour < 18) {
        backgroundIndex = 3;
    } else if (hour >= 18 && hour < 22) {
        backgroundIndex = 4;
```

```
    } else { // 22-2 点
        backgroundIndex = 5;
    }
    backgroundLayout.setBackgroundResource(hourBackgrounds[backgroundIndex]);
}
```

对应效果如下：

对应布局activity_time.xml 及布局如下：

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/clock_background"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="16dp">

    <TextView
        android:id="@+id/timeTextView"
        android:layout_width="204dp"
        android:layout_height="68dp"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_marginStart="92dp"
        android:layout_marginTop="154dp"
        android:text="12:00:00"
        android:textColor="@color/white"
        android:textSize="72sp"
        android:textStyle="bold" />
    <TextView
        android:id="@+id/dateTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/timeTextView"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="20dp"
        android:text="2024 年 5 月 1 日"
        android:textColor="@color/white"
        android:textSize="24sp" />
    <TextView
        android:id="@+id/weekdayTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/dateTextView"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="10dp"
        android:text="星期三"
        android:textColor="@color/white"
        android:textSize="20sp" />

</RelativeLayout>
```



三、 技术实现

3.1 定时器机制

关键代码 startClock() 如下：

```
private void startClock() {

    timer = new Timer(); //  创建一个 Timer 对象
    timer.schedule(new TimerTask() {
        @Override
        public void run() {
            runOnUiThread(() -> updateClock());   //  在主线程执行
        }
    }, 0, 10000); //  每 10 秒更新一次
}
```

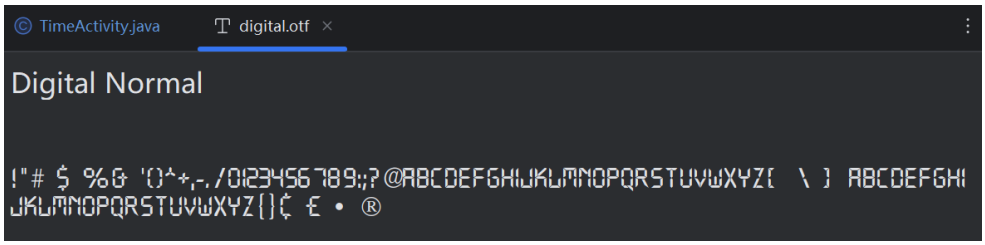
UpdateClock() 的代码如下：

```
private void updateClock() {
    Calendar = Calendar.getInstance();
    Date now = calendar.getTime();

    // (HH:mm:ss)
    SimpleDateFormat timeFormat = new SimpleDateFormat("HH:mm:ss",
Locale.getDefault());
    timeTextView.setText(timeFormat.format(now));
```

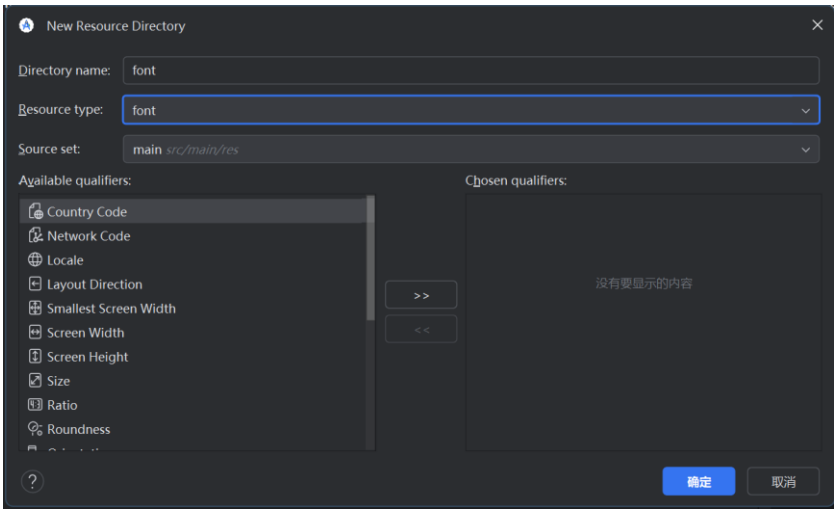
```
        //   (yyyy 年 MM 月 dd 日)
        @SuppressLint("SimpleDateFormat") SimpleDateFormat dateFormat = new
SimpleDateFormat("yyyy 年 MM 月 dd 日");
        dateTextView.setText(dateFormat.format(now));

        // (星期 x)
        SimpleDateFormat weekdayFormat = new SimpleDateFormat("E", Locale.CHINESE);
        weekdayTextView.setText(weekdayFormat.format(now));

        int hour = calendar.get(Calendar.HOUR_OF_DAY);
        if (hour != currentHour) {
            currentHour = hour;
            updateBackground(hour);
        }
}
```

3.2 digital 字体的实现

首先我从网络上下载了 digital.otf 的字体文件:



发现此字体只包含英文符号和数字，故只能将时间文本设置字体为当前字体，在 resource 文件夹下新建了 font 资源文件夹，将当前 otf 文件添加到这里:



然后在 initView() 中初始化字体:

```
// 设置数字字体
Typeface digitalFont;

try {
    digitalFont = androidx.core.content.res.ResourcesCompat.getFont(this, R.font.digital);
} catch (Exception e) {
    digitalFont = Typeface.MONOSPACE;
}
timeTextView.setTypeface(digitalFont);
```

四、 课程小结

本次项目不仅巩固了我的 Java 和 Android 开发知识，还让我深刻体会到用户体验设计的重要性。从需求分析到代码实现，每一步都需要细致考虑用户的实际需求和技术可行性。通过这次实践，我对 Android 开发的完整流程有了更清晰的认识，为后续复杂应用的开发打下了坚实基础。