Day9 3月8日

软件51 庞建业 2151601012

Vue.js学习

> Vue.js（读音 /vjuː/, 类似于 view） 是一套构建用户界面的 渐进式框架。与其他重量级框架不同的是，Vue 采用自底向上增量开发的设计。Vue 的核心库只关注视图层，并且非常容易学习，非常容易与其它库或已有项目整合。另一方面，Vue 完全有能力驱动采用单文件组件和 Vue 生态系统支持的库开发的复杂单页应用。
>
> Vue.js 的目标是通过尽可能简单的 API 实现响应的数据绑定和组合的视图组件。

Vue.js就是一个用于搭建类似于网页版知乎这种表单项繁多，且内容需要根据用户的操作进行修改的网页版应用。

单页应用程序（SPA）

顾名思义，单页应用一般指的就是一个页面就是应用，当然也可以是一个子应用，比如说知乎的一个页面就可以视为一个子应用。单页应用程序中一般交互处理非常多，而且页面中的内容需要根据用户的操作动态变化。

声明式渲染

Vue.js 的核心是一个允许采用简洁的模板语法来声明式地将数据渲染进 DOM 的系统：

```
<div id="app">
  {{ message }}
</div>

var app = new Vue({
  el: '#app',
  data: {
    message: 'Hello Vue!'
  }
})
```

响应式的数据绑定

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>vue.js测试 - 代码之美专栏</title>
    <!-- author:昌维 代码之美 https://zhuanlan.zhihu.com/codes -->
    <script src="https://unpkg.com/vue/dist/vue.js"></script>
</head>
<body>
    <div id="app">
        <input type="text" name="" value="" placeholder="在这里输入文字，下面会跟着变化" v-
model="message">
        <hr>
        <p>{{ message }}</p>
    </div>
```

```
    <script type="text/javascript">
        var app = new Vue({
          el: '#app',
          data: {
            message: 'Hello Vue!'
          }
        })
    </script>
</body>
</html>
```

p标签里面通过{{ message }}这个写法与input标签中的value绑定在了一起，其中变化，另外一个和它绑定的数据就跟着变化。

vue.js会自动响应数据的变化情况，并且根据用户在代码中预先写好的绑定关系，对所有绑定在一起的数据和视图内容都进行修改。而这种绑定关系，在图上是以input 标签的v-model属性来声明的，因此在别的地方可能也会看到有人粗略的称vue.js为声明式渲染的模版引擎。

组件化开发

在面向对象编程中，我们可以使用面向对象的思想将各种模块打包成类或者把一个大的业务模块拆分成更多更小的几个类。在面向过程编程中，我们也可以把一些大功能拆分成许多函数，然后分配给不同的人来开发。

引入了组件化开发的思想:

Vue.js通过组件，把一个单页应用中的各种模块拆分到一个一个单独的组件（component）中，我们只要先在父级应用中写好各种组件标签（占坑），并且在组件标签中写好要传入组件的参数（就像给函数传入参数一样，这个参数叫做组件的属性），然后再分别写好各种组件的实现（填坑），然后整个应用就算做完了。

Virtual DOM

Virtual DOM是虚拟DOM的英文，简单来说，他就是一种可以预先通过JavaScript进行各种计算，把最终的DOM操作计算出来并优化，由于这个DOM操作属于预处理操作，并没有真实的操作DOM，所以叫做虚拟DOM。最后在计算完毕才真正将DOM操作提交，将DOM操作变化反映到DOM树上。

Andriod前端页面实践

QuickSideBar

帮助快速查阅对应分组的侧边栏，可以配合任意列表

DividerDecoration.java

```
/**
 * RecyclerView分割条
 */
public class DividerDecoration extends RecyclerView.ItemDecoration {

  private static final int[] ATTRS = new int[]{
      android.R.attr.listDivider
  };

  public static final int HORIZONTAL_LIST = LinearLayoutManager.HORIZONTAL;
```

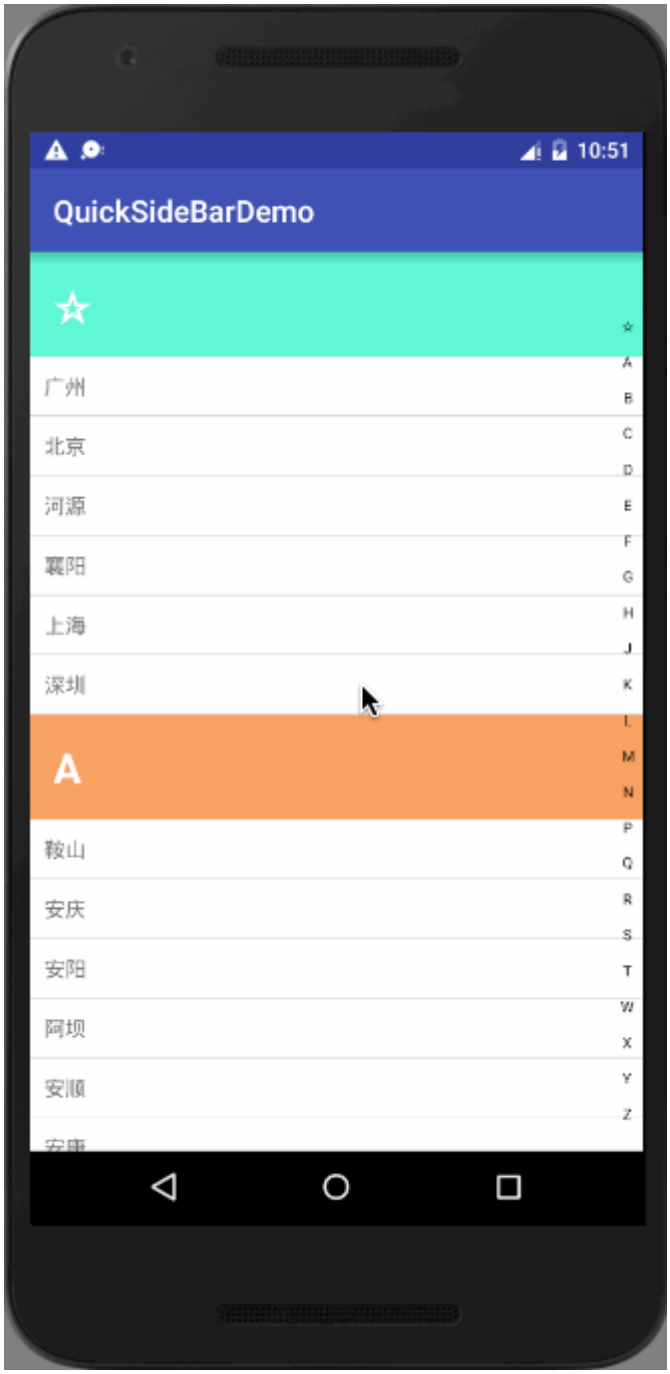
```
public static final int VERTICAL_LIST = LinearLayoutManager.VERTICAL;

private Drawable mDivider;

public DividerDecoration(Context context) {
  final TypedArray a = context.obtainStyledAttributes(ATTRS);
  mDivider = a.getDrawable(0);
  a.recycle();
}

private int getOrientation(RecyclerView parent) {
  LinearLayoutManager layoutManager;
  try {
    layoutManager = (LinearLayoutManager) parent.getLayoutManager();
  } catch (ClassCastException e) {
    throw new IllegalStateException("DividerDecoration can only be used with a " +
        "LinearLayoutManager.", e);
  }
  return layoutManager.getOrientation();
}

@Override
public void onDraw(Canvas c, RecyclerView parent, RecyclerView.State state) {
  super.onDraw(c, parent, state);

  if (getOrientation(parent) == VERTICAL_LIST) {
    drawVertical(c, parent);
  } else {
    drawHorizontal(c, parent);
  }
}

public void drawVertical(Canvas c, RecyclerView parent) {
  final int left = parent.getPaddingLeft();
  final int right = parent.getWidth() - parent.getPaddingRight();
  final int recyclerViewTop = parent.getPaddingTop();
  final int recyclerViewBottom = parent.getHeight() - parent.getPaddingBottom();
  final int childCount = parent.getChildCount();
  for (int i = 0; i < childCount; i++) {
    final View child = parent.getChildAt(i);
    final RecyclerView.LayoutParams params = (RecyclerView.LayoutParams) child
        .getLayoutParams();
    final int top = Math.max(recyclerViewTop, child.getBottom() + params.bottomMargin);
    final int bottom = Math.min(recyclerViewBottom, top + mDivider.getIntrinsicHeight());
    mDivider.setBounds(left, top, right, bottom);
    mDivider.draw(c);
  }
}

public void drawHorizontal(Canvas c, RecyclerView parent) {
  final int top = parent.getPaddingTop();
  final int bottom = parent.getHeight() - parent.getPaddingBottom();

  final int recyclerViewLeft = parent.getPaddingLeft();
```

```
    final int recyclerViewRight = parent.getWidth() - parent.getPaddingRight();
    final int childCount = parent.getChildCount();
    for (int i = 0; i < childCount; i++) {
      final View child = parent.getChildAt(i);
      final RecyclerView.LayoutParams params = (RecyclerView.LayoutParams) child
          .getLayoutParams();
      final int left = Math.max(recyclerViewLeft, child.getRight() + params.rightMargin);
      final int right = Math.min(recyclerViewRight, left + mDivider.getIntrinsicHeight());
      mDivider.setBounds(left, top, right, bottom);
      mDivider.draw(c);
    }
  }

  @Override
  public void getItemOffsets(Rect outRect, View view, RecyclerView parent, RecyclerView.State
state) {
    super.getItemOffsets(outRect, view, parent, state);
    if (getOrientation(parent) == VERTICAL_LIST) {
      outRect.set(0, 0, 0, mDivider.getIntrinsicHeight());
    } else {
      outRect.set(0, 0, mDivider.getIntrinsicWidth(), 0);
    }
  }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <android.support.v7.widget.RecyclerView
        android:id="@+id/recyclerView"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
    <!-- 这个是浮动的提示 , 配合字母栏实现放大浮动提示滑动到哪个字母-->
    <!--下面的自定义属性都是默认的,可以不写-->
    <!--app:sidebarBackgroundColor 浮动框颜色-->
    <!--app:sidebarTextColor 字母颜色-->
    <!--app:sidebarTextSize 字母尺寸-->
    <com.bigkoo.quicksidebar.QuickSideBarTipsView
        android:id="@+id/quickSideBarTipsView"
        android:layout_width="@dimen/height_quicksidebartips"
        android:layout_height="match_parent"
        android:layout_toLeftOf="@+id/quickSideBarView"
        app:sidebarBackgroundColor="@color/colorPrimary"
        app:sidebarTextColor="@android:color/white"
        app:sidebarTextSize="@dimen/textSize_quicksidebartips" />
    <!-- 这个是字母栏的提示 -->
    <!--下面的自定义属性都是默认的,可以不写-->
    <!--app:sidebarItemHeight 每个字母的高度-->
```

```
    <!--app:sidebarTextColor 正常状态下字母颜色-->
    <!--app:sidebarTextColorChoose 选中了的字母颜色-->
    <!--app:sidebarTextSize 正常状态字母尺寸-->
    <!--app:sidebarTextSizeChoose 选中字母尺寸-->
    <com.bigkoo.quicksidebar.QuickSideBarView
        android:id="@id/quickSideBarView"
        android:layout_width="20dp"
        android:layout_height="match_parent"
        android:layout_alignParentRight="true"
        android:layout_marginTop="45dp"
        app:sidebarItemHeight="@dimen/height_quicksidebaritem"
        app:sidebarTextColor="@android:color/black"
        app:sidebarTextColorChoose="@color/colorPrimary"
        app:sidebarTextSize="@dimen/textSize_quicksidebar"
        app:sidebarTextSizeChoose="@dimen/textSize_quicksidebar_choose" />
</RelativeLayout>
```



CalendarListView

calendarlistview 选取提供了一个 API 10+日历日期的简便方法

只需要布局中添加 datepickerview 无需定制。

使用控件 RecycleView 实现的日期选择器，可以选择时间段。

在 gradle 中

```
dependencies {

    compile 'com.github.traex.calendarlistview:library:1.2.2'

}
```

1.在你的布局 XML 声明一个 `DayPickerView`

2.在 Activity 或 Fragment 中引入 `DatePickerController`,然后你需要设置 `getMaxYear` and > `onDayOfMonthSelected`. `getMaxYear` 是设置选择器的最大年数 `onDayOfMonthSelected` 是当你选择了某个新日期时的回调

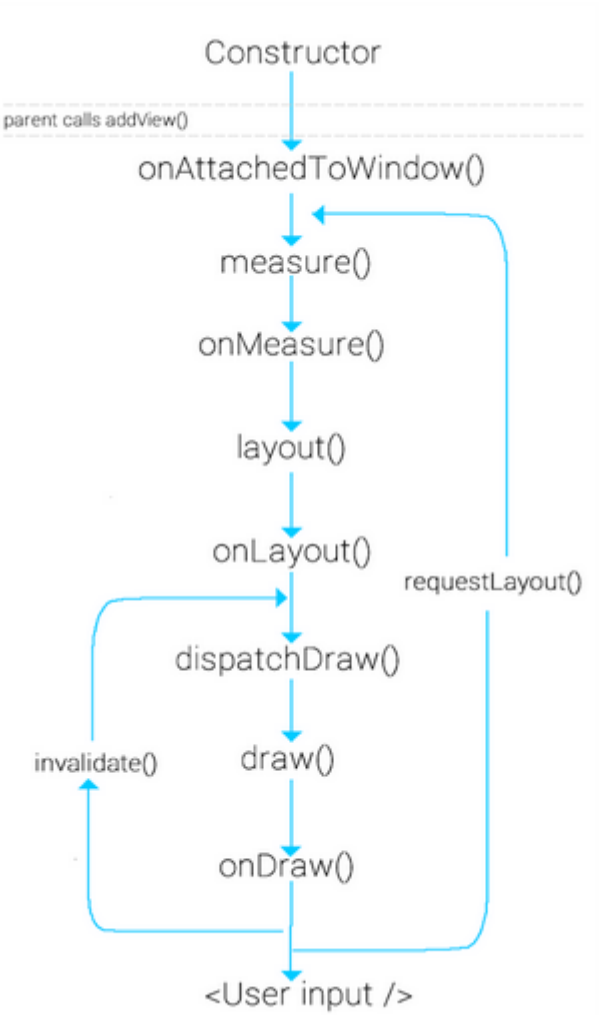
java

```
@Override

public int getMaxYear()

{

    return 2015;

}

@Override

public void onDayOfMonthSelected(int year, int month, int day)

{

    Log.e("Day Selected", day + " / " + month + " / " + year);

}
```

定制方法:

- app:colorCurrentDay [color def:#ff999999] --> 当前日期固定为粗体，但你可以定制颜色
- app:colorSelectedDayBackground [color def:#E75F49] --> 点击的日期的背景圆形或圆角矩形的颜色
- app:colorSelectedDayText [color def:#fff2f2f2] --> 已选择天数的文字颜色
- app:colorPreviousDay [color def:#ff999999] --> 当前日期之前的颜色
- app:colorNormalDay [color def:#ff999999] --> 默认日期的颜色
- app:colorMonthName [color def:#ff999999] --> 月名和年名的默认颜色
- app:colorDayName [color def:#ff999999] --> 日期名字的默认颜色
- app:textSizeDay [dimension def:16sp] --> 日期数字的文本大小
- app:textSizeMonth [dimension def:16sp] --> 月份的文本大小
- app:textSizeDayName [dimension def:10sp] --> 日期名的文本大小
- app:headerMonthHeight [dimension def:50dip] --> 顶部月份栏的高度
- app:drawRoundRect [boolean def:false] --> 选中天数使用圆角矩形而不是使用圆形
- app:selectedDayRadius [dimension def:16dip] --> 如果使用圆角矩形，设置圆角矩形的半径
- app:calendarHeight [dimension def:270dip] --> 月或行的行距
- app:enablePreviousDay [boolean def:true] --> 启用已过期的天数

- app:currentDaySelected [boolean def:false] --> 默认选中当前日期
- app:firstMonth [enum def:-1] --> 默认起始月份
- app:lastMonth [enum def:-1] --> 默认结束月份




```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
                xmlns:calendar="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <com.andexert.calendarlistview.library.DayPickerView
        android:id="@+id/pickerView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        calendar:drawRoundRect="true"/>

</RelativeLayout>
```

CollectionView

支持异步加载子列表的 ExpandableListView，包括CollectionView可以显示小标题的列表

```
<com.ericliu.asyncexpandablelist.CollectionView
        android:id="@+id/collectionView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>
```

```
 int groupOrdinal = 0; // groupOrdinal dictates the sequence of groups to be displayed in the
list
      CollectionView.InventoryGroup<String, News> group1 = inventory.newGroup(groupOrdinal);

      // creating objects to be populated into the list.
      News news1 = new News();
      ...
      News news2 = new News(); ......
      .......

      // set the header item, in this case, it is simply a String.
      group1.setHeaderItem("Top Stories");
      // add items under this header.
      group1.addItem(news1);
      group1.addItem(news2);
      group1.addItem(news3);
        ....
```

```
 mCollectionView.updateInventory(inventory);
```