flutter 学习二

1. 课前复习

- App结构和导航组件
  - MaterialApp应用组件
  - Scaffold脚手架组件
  - Flutter主题

2. 课堂目标

- 手势
- 可滚动组件
  - ListView
  - GridView
  - SingleChildScrollView

3. 知识点

手势

手势：在flutter中手势主要分为三种：轻击、拖动和缩放

如果想给一个widget具有手势，则：

```
GestureDetector(
  onTap:(()=>{
    print('GestureDetector action')
  }),
  child: Text('Align'),
)
```

可滚动组件

ListView

- 常用属性
  - scrollDirection：滚动方向，`horizontal`和`vertical`，默认是垂直方向上滚动；
  - physics：列表滚动至边缘后继续拖动的物理效果，安卓是一个波纹状（对应`ClampingScrollPhysics`），iOS是有一个回弹的弹性效果（对应`BouncingScrollPhysics`），如果想不同的平台上呈现各自的效果可以使

用`AlwaysScrollableScrollPhysics`，它会根据不同平台自动选用各自的物理效果。如果你想禁用在边缘的拖动效果，可以使用`NeverScrollableScrollPhysics`；
    - shrinkWrap：该属性将决定列表的长度是否仅包裹其内容的长度。当`ListView`嵌在一个无限长的容器组件中时，`shrinkWrap`必须为true，否则会给出警告；
    - padding：内间距
    - itemExtent：子元素长度。当列表中的每一项长度是固定的情况下可以指定该值，有助于提高列表的性能（因为它可以帮助`ListView`在未实际渲染子元素之前就计算出每一项元素的位置）
    - cacheExtent：预渲染区域长度，`ListView`会在其可视区域的两边留一个`cacheExtent`长度的区域作为预渲染区域（对于`ListView.build`或`ListView.separated`构造函数创建的列表，不在可视区域和预渲染区域内的子元素不会被创建或会被销毁）
    - children：容纳子元素的组件数组
- 创建listView的方式
    - 方式一：ListView()（特点：代码简洁，对于小批量固定数据可以考虑使用，但是数据量大则性能不好，因为这种方式创建类似于RN中的scrollView，即使还没有出现在屏幕中但仍然会被ListView所创建，这将是一项较大的开销，使用不当可能引起性能问题甚至卡顿）
    - 方式二：ListView.build()：绝大多数列表类的需求都可以用`ListView.build`构造函数来解决问题
        - `itemCount`: 列表中元素的数量
        - `itemBuilder`: 子元素的渲染方法，允许自定义子元素组件（等同于`rn`中`FlatList`组件的`renderItem`属性）
    - 方式三：ListView.separated()：列表子项之间需要`分割线`，此时可以考虑用此方法
        - `separatorBuilder`:构造分割线

GridView：网格布局，适用于多行多列的情况

- 方式一：GridView.count

> 这种方式如果指定单个widget的宽高是不会起作用的，因为这里已经指定了每一行分成几列以及宽高比，还有边距等等

```
GridView.count(
      //水平子Widget之间间距
      crossAxisSpacing: 10.0,
      //垂直子Widget之间间距
      mainAxisSpacing: 30.0,
      //GridView内边距
      padding: EdgeInsets.all(10.0),
      //一行的Widget数量
      crossAxisCount: 2,
      //子Widget宽高比例
      childAspectRatio: 2.0,
      //子Widget列表
      children: widgetList(),
 );
```

- 方式二：GridView.builder

```
GridView.builder(
    itemCount: datas.length,
    //SliverGridDelegateWithFixedCrossAxisCount 构建一个横轴固定数量Widget
    gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
      //横轴元素个数
        crossAxisCount: 3,
        //纵轴间距
        mainAxisSpacing: 20.0,
        //横轴间距
        crossAxisSpacing: 10.0,
        //子组件宽高长度比例
        childAspectRatio: 1.0),
    itemBuilder: (BuildContext context, int index) {
      //Widget Function(BuildContext context, int index)
      return getItemContainer(datas[index]);
    });
```

- 方式三：GridView.builder（SliverGridDelegateWithMaxCrossAxisExtent）

```
GridView.builder(
      itemCount: datas.length,
      itemBuilder: (BuildContext context, int index) {
        return getItemContainer(datas[index]);
      },
      gridDelegate: SliverGridDelegateWithMaxCrossAxisExtent(
        //单个子Widget的水平最大宽度
        maxCrossAxisExtent: 200,
        //水平单个子Widget之间间距
        mainAxisSpacing: 20.0,
        //垂直单个子Widget之间间距
        crossAxisSpacing: 10.0
      ),
    );
```

> 对于`SliverGridDelegateWithMaxCrossAxisExtent`而言，水平方向元素个数不再固定，其水平个数也就是有几列，由`maxCrossAxisExtent`和屏幕的宽度以及`padding`和`mainAxisSpacing`等决定。

- 方式四：GridView.custom

```
@override
  Widget build(BuildContext context) {
    List<String> datas = getDataList();
    return GridView.custom(
        gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
            crossAxisCount: 3, mainAxisSpacing: 10.0, crossAxisSpacing:
20.0, ),
        childrenDelegate: SliverChildBuilderDelegate((context, position) {
          return getItemContainer(datas[position]);
        }, childCount: datas.length));
  }
```

SingleChildScrollView：可滑动的view，类似于原生和RN中的ScrollView，其只能包含一个子元素

常用属性： **scrollDirection**：滚动方向，默认是垂直 **reverse**：是否按照阅读方向相反的方向滑动。 **padding**：填充距离 **primary**：是否使用 widget 树中默认的 PrimaryScrollController 。当滑动方向为垂直方向（scrollDirection值为Axis.vertical）并且controller没有指定时，primary默认为true **physics**：此属性接受一个ScrollPhysics对象，它决定可滚动Widget如何响应用户操作，比如用户滑动完抬起手指后，继续执行动画；或者滑动到边界时，如何显示。默认情况下，Flutter会根据具体平台分别使用不同的ScrollPhysics对象，应用不同的显示效果，如当滑动到边界时，继续拖动的话，在iOS上会出现弹性效果，而在Android上会出现微光效果。如果你想在所有平台下使用同一种效果，可以显式指定，Flutter SDK中包含了两个ScrollPhysics的子类可以直接使用：ClampingScrollPhysics→Android下微光效果 / BouncingScrollPhysics→iOS下弹性效果 **controller**：此属性接受一个ScrollController对象。ScrollController的主要作用是控制滚动位置和监听滚动事件 **child**：子元素

Table：表格组件

常用属性：

- **columnWidths**：每一列的宽度
- **defaultColumnWidth**：默认的每一列宽度值，默认情况下均分
- **textDirection**：文字方向，一般无需考虑
- **border**：表格边框
- **defaultVerticalAlignment**：每一个cell的垂直方向的alignment。 top：被放置在的顶部； middle：垂直居中； bottom：放置在底部； baseline：文本baseline对齐； fill：充满整个cell
- **textBaseline**：defaultVerticalAlignment为baseline的时候，会用到这个属性

```
Table(
  children: items,
  columnWidths: <int, TableColumnWidth>{
    0: FixedColumnWidth(100.0),
    1: FixedColumnWidth(40.0),
    2: FixedColumnWidth(100.0),
  },
  border: TableBorder.all(
      color: Color(0xffdddddd), width: 1.0, style: BorderStyle.solid),
)
```

4. 总结

- 手势
- 列表

5. 口令

不用背，学规律

6. 作业 && 答疑

完成如下界面

. 下节课内容

- 路由的使用
- 数据持久化
- 网络控件的使用