Flutter

> 以下为对Flutter官网的学习总结，如果你想快速掌握知识点，或者想复习一下官网学习的内容，那么值得看看。

用户界面

widgets介绍

- Flutter一切都是widget，包括设置padding的container。
- 几乎所有widget都通过build方法声明其UI
- StatelessWidget用于固定样式的widget，StatefulWidget用于根据数据变化的widget。
- StatefulWidget通过createState关联私有的State对象，并通过setState()方法更新数据并通知UI变化。
- 更新UI时Flutter会通过比较前后widget树来计算差异，widget只是保存了样式信息，它的重建可以考虑是轻量级的。widget树会对应到element树，并通过element树创建Render树。相同类型widget会重用element和render对象。
- State对象的生命周期跨越其对应的widget对象build方法，比widget本身生命周期要长
- State调用流程大致为initState -> build -> dispose，可以在initState做初始化操作，在dispose中做清理操作
- didChangeDependencies会在initState和build之间调用，当父widget有InheritedWidget变化时也会被调用
- InheritedWidget可用于在widget树中给子widget共享数据，通常通过of方法调用context.inheritFromWidgetOfExactType返回拥有共享数据的InheritedWidget对象
- key控制widget重建时与哪些其他widget进行匹配，从而保持正确的state状态，一般用在widget的添加删除或者重排序中控制widget重用
- key分为Local key（value key表示根据某个值判断、Object key表示根据某个对象判断、Unique key表示每个widget都不一样） 和Global key(表示不同页面的widget共享),

构建layouts

Flutter中的layouts

- 可以通过Row和Column构建复制页面
- mainAxisAlignment控制主轴对齐方式，crossAxisAlignment控制次轴对齐方式
- 使用Expanded widget来fit window，flex来指定比例
- 将布局widget赋值给变量，通过变量组合布局减少层级嵌套
- 使用Container设置margin、border、pandding和背景
- GridView.extend中maxCrossAxisExtent设置每个item的最大宽度，mainAxisSpacing设置主轴item之间的间隔，crossAxisSpacing设置次轴item之间的间隔，childAspectRatio设置item宽高比例
- GridView.builder用于数量较多的item展示，仅加载当前可见的部分，GridView.count用于加载少量固定数目的item并指定每行item格式，GridView.extend用于加载少量固定item并指定每行item最大宽度
- GridView中通过SliverGridDelegate控制子widget如何布局，通过SliverChildDelegate来获取子widget，可以通过自定义来实现自由或者叠加布局。
- GridView和ListView都继承自BoxScrollView
- 大量数据需使用ListView.builder并在itemBuilder回调中创建并提供widget；如果列表的item样式可以提前构建则可以直接使用new ListView；ListView.separated除了itemBuilder之外还有个separatorBuilder用来定义分隔线样式；ListView.custom通过提供自定义的SliverChildDelegate来
- Stack用于widget的堆叠，可以做渐变的图片阴影
- Card内部内容不能够滚动，可以自定义圆角和阴影大小
- ListTitle是方便构建至多三行文字加上前后图标的列表item widget

layout使用例子

- 使用Expanded widget占满剩余空间，子widget设置CrossAxisAlignment.start表示从前开始
- Text softwrap控制是否需要自动换行
- 修改pubspec.yaml设置assets目录，例如：flutter: assets: [images/]
- Image.asset中设置fit:BoxFit.cover 表示图片应该以最小的大小占满box空间
- 使用ListView代替Column保证小屏幕手机中空间可以滚动

创建自适应UI应用

- 使用LayoutBuilder的BoxConstraints获取当前widget的宽高比例从而调整子widget布局
- 使用MediaQuery.of()获取屏幕宽高和旋转方向等设备信息从而控制整体布局样式
- AspectRatio控制子widget的宽高比例
- CustomSingleChildLayout、CustomMultiChildLayout将子widget的布局委托给ChildLayoutDelegate进行控制

- FittedBox：当子widget比父widget大时，通过FittedBox可以设置子widget的缩放方式
- FractionallySizedBox可以设置子widget占据其空间的宽高百分比
- MediaQueryData中padding指代周边有多少不能绘制的区域不计算被键盘等遮挡的区域，viewPadding指的是周边有多少不能被绘制的区域不受键盘等遮挡影响，viewInsert表示周边有多少区域被键盘等遮挡了
- OrientationBuilder获取屏幕是否旋转

Box constraints

- 有三种box，分别是无限扩展例如Center或者ListView、子widget决定例如Trnasform和Opacity、固定大小例如Image和Text
- 类似于当一个竖向的ListView嵌套进了一个横向的Lis他View，会造成无界约束状态（Unbounded constraints），这种状态会使得子widget可以在两个方向无限扩展导致错误
- Flex boxs指代Row和Column，表示当其处于一个有节的区域会不断扩展至给定大小，当其处于一个无界区域会适应他的子widget大小。
- 如果将Flex box置于类似于ListView的widget中，那么flex box中不能有类似于Expanded的widget，这会导致类似于Expanded的widget无限扩大造成错误
- Column的宽度和Row的高度不能设置为无界的，否则他们的子widget将无法布局

Android闪屏

- 设置应用打开闪屏的设置与原生方式一样，都是给第一个打开的activity设置主题
- 设置name为io.flutter.embedding.android.NormalTheme的meta-data来定义正常主题，这样就会使得页面从启动的主题转变为正常的主题。
- 在Android Activity启动后，还需要初始化Dart isolate，这段时间可以再设置闪屏
- 设置Flutter的闪屏有两种方法，一种是设置展示一个drawable，可以在Activity的Manifest中设置name为io.flutter.embedding.android.SplashScreenDrawable的meta-data并指定drawable资源，或在Fragment中重写provideSplashScreen方法返回一个DrawableSplashScreen对象。第二种方法是实现SplashScreen接口，通过createSplashView提供自定义闪屏view，并通过transitionToFlutter方法标记闪屏view动画是否完成。

加入互动逻辑

- 可交互的widget有三点，一是有两个类，分别继承StatefulWidget和State，二是State类中拥有可变的状态和build方法，三是当状态变化，调用setState()方法对widget进行重绘。
- 将Text放在SizedBox中可以防止当文字变化时由于宽度变化带来的位置抖动
- 当调用setState({})方法时，会先执行lambda逻辑，然后调用_element.markNeedsBuild()标记当前element为dirty状态并在下一帧根据修改后的状态进行重绘

- 有三种常见的管理状态方法，分别是：widget自己管理自己的状态、父widget管理状态、混合前两种方式
- 如果状态是用户数据，那么最好在父widget管理。如果状态是与界面效果有关的例如动画，那么最好在widget自身内管理状态。如果不确定最好先在父widget中管理，因为大多数情况外层需要对状态数据进行处理并更新子widget，外层处理状态也有利于子widget保持整洁。当widget既包含用户状态又包含外部不关注的自身界面效果状态则使用混合状态管理模式。
- 对于必须传入的参数使用@require注解

添加assets和图片

- 在pubspec.yaml中声明assets文件夹的路径声明，如果需要添加子文件夹的话需要单独列出
- 声明assets时会同时查找其定义的子文件夹是否有同名的文件，如果有的话会把同名的文件同时引入，这是为了方便引入不同分辨率的图片资源
- 使用DefaultAssetBundle.of(context).load()或loadString()方法加载asset文本资源，其中context最好使用当前widget的BuildContext，这有利于父widget在测试或者本地化时在运行时替换不同的AssetBundle。
- 当不能获取widget context的地方，可以使用rootBundle来加载文本资源
- 对于图片资源，可以使用相同的图片命名并放在2.0x和3.0x文件夹中，不同dp/px比例的手机会自动选用合适大小的资源
- 使用AssetImage加载图片会自动选择对应分辨率的图片，如果需要加载不同package的图片，需要在AssetImange中指定package
- 对于不在同一个package的图片资源，也需要在pubspec.yaml文件中定义，例如需要引用package为fancy_backgrounds的图资源，需要在当前的pubspec.yaml中定义assets路径为packages/fancy_backgrounds/xxx(图片在fancy_backgrounds中libs目录下的相对位置)
- 在Android中使用flutter的asset资源，使用PluginRegistry.Registrar.lookupKeyForAsset()方法获取key，并使用AssetManager.openFd(key)方法获取AssetFileDescriptor
- 在iOS中使用flutter的asset资源，可以使用registrar lookupKeyForAsset或者key，然后通过mainBundle pathForResource:key ofType或者asset路径。如果使用了ios_platform_images插件，那么可以直接使用OC中的UIImage flutterImageWithName或者Swift中的UIImage.flutterImageNamed获取。
- flutter中使用iOS的图片可以使用ios_platform_images插件中的IosPlatformImages.load方法
- 启动页会在Flutter绘制第一帧的时候被替换，如果在main方法中不调用runApp方法，那么启动页将一直展示。
- 加入启动页的方式需要使用Android和iOS的本身的方式加入。

页面导航

导航到新的页面和返回

- route在安卓中相当于Activity，在iOS中相当于ViewController，在Flutter中，route表示的只是一个widget
- 页面导航的步骤：创建两个route，使用Navigator.push()导航到第二个route，使用Navigator.pop()返回到上一个route
-

使用具名的路由进行导航

传递数据到新的页面

使用具名路由传递参数

从页面中返回数据

在两个页面中共享元素动画

学习资源

- Flutter samples
- Flutter YouTube playlist
- The Mahogany Staircase - Flutter's Layered Design