一、Widget简介

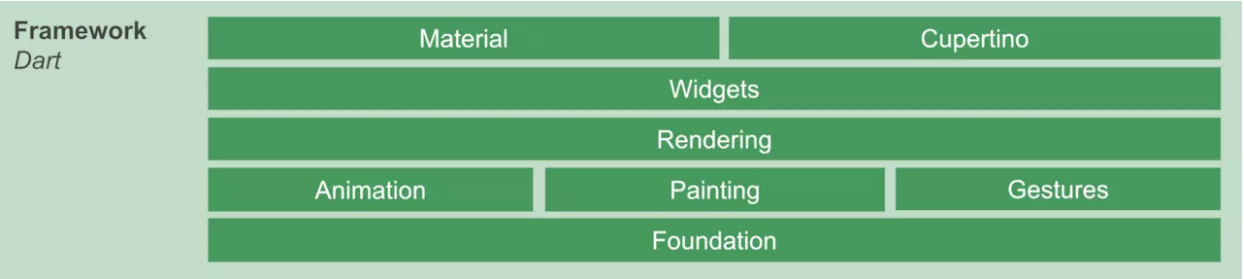
Flutter中的Widget相当于Android中的View，IOS中的UIView

Flutter使用Widget来构建UI，类似于Android中使用基础组件来构建UI。

1、Flutter的渲染

Flutter的渲染类似于React的框架，在Widget发生改变需要更新界面时，框架会计算从上一个状态到下一个状态所需的最小改变，然后将这些 改变去应用到界面上，从而更新界面，即热更新

2、Flutter的Framework



从上到下，依次为:

（1）Material与Cupertino

Material是为Android设计的UI风格，而Cupertino则是IOS的UI风格。Flutter中存在这两种UI风格可以让Flutter适应于不同的平台，从而 使Flutter获得与Native UI一样的使用体验

（2）Widgets

Widgets则是Flutter为开发人员提供的UI组件，便于创建UI

（3）Animation、Painting、Gestures

动画、绘制、手势是Flutter提供的基本功能，平时很少用到，类似于Android中的自定义View时使用，通常用这些来实现一些复杂的组件

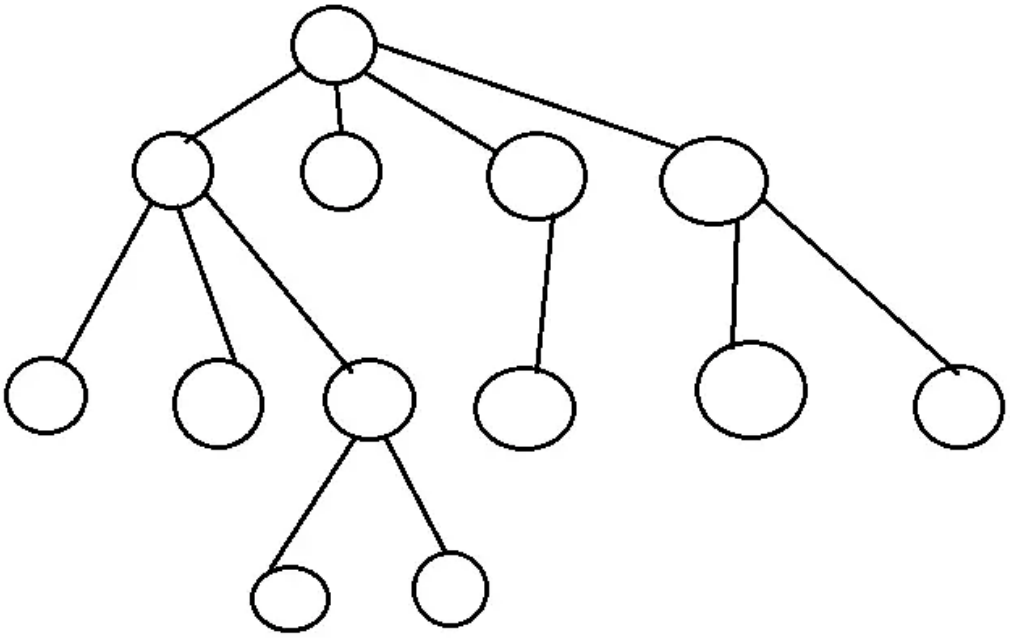
（4）Foundation

Foundation是Flutter提供的基础类库

二、Widget树

Widget组合的结构是树，因此也成为Widget树。

Widget树的结构如下:



1、父Widget和子Widget

在Widget树中，widget有包含和被包含关系

父Widget：包含有子Widget的Widget称为父Widget

子Widget：被父Widget包含的就是子Widget

2、根Widget

根Widget也称为Root Widget。

如下代码：

```
void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
 // This widget is the root of your application.
 @override
 Widget build(BuildContext context) {
 return MaterialApp(
 title: 'Flutter Demo',
 theme: ThemeData(
  primarySwatch: Colors.blue,
  ),
  home: MyHomePage(title: 'Flutter Demo Home Page'),
  );
  }
}
```

这里的Root Widget是MaterialApp，而不是MyApp，因为MyApp只是用来封装一下。

在Flutter中默认根Flutter会充满屏幕

在Flutter中根Widget只能是以下三个：

- WidgetsApp
- MaterialApp
- CupertinoApp

3、Widget如何判断Widget是否改变

（1）Widget的标识符Key

Key是作为判断Widget是否改变的判断条件之一。

通常key分为两类：

1> Local Key（局部key）

在同一父Widget中的Widget，Key值在需要在父Wiget包含的子Wiget范围内唯一，这类Key称为局部Key

但是LocalKey又有不同的实现，常见如下：

- ObjectKey

将对象作为Key值

- ValueKey

将特定类型的值作为Key的值

- UniqueKey

使用UniqueKey自己的对象作为Key的值

2>Global Key（全局key）

全局Key在整个app中唯一

Global key的实现也不同，常见如下：

- LabeledGlobalKey

该Key值用于调试，不会用来比较Widget是否变化

- GlobalObjectKey

将对象作为Key值

一般我们不需要使用Key，当页面比较复杂时，可以使用Key来提高渲染性能

（2）如何判断Widget有没有变化

主要在于diff计算。因为不一定使用key来标识Widget，因此这里的判断又分为两种情况：

1>默认情况下（Widget没有设置Key）

在没有设置Key值时，Flutter会通过Widget的runTimeType和显示顺序来判断Widget是否发生变化。

其中runTimeType是Widget的类型

2>设置Key

在设置Key值时，会根据Key值和runTimeType来判断Widget是否发生变化。

每次刷新UI时，都会重新构建Widget树，通过对比构建前后的Widget，计算出变化的部分，更新时只更新变化的Widget即可

而这里判断变化就是还要通过标识符来进行判断

四、Widget分类

因为渲染很耗性能，为了提高Flutter的帧率，就需要减少不必要的的渲染，所以Flutter根据UI是否改变，分为两类：

- StatefulWidget

StatefulWidget是UI可以变化的Widget，创建完毕后还可以改变

- StatelessWidget

StatelessWidget是UI不可以变化的Widget，创建完毕后就不再改变。

五、Widget大全

Widget一共分为14类，如下：

1. Accessibility
2. Animation and Motion
3. Assets, Images, and Icons
4. Async
5. Basics
6. Cupertino (iOS-style widgets)
7. Input
8. Interaction Models
9. Layout
10. Material Components
11. Painting and effects
12. Scrolling
13. Styling
14. Text

例如常见的布局、UI组件、动画等