

Flutter 开发利器 —— Hot Reload

前面已经将 Flutter 运行起来了，然后打开 main.dart，看到如下的代码：

```

class MyApp extends StatelessWidget {
  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        // This is the theme of your application.
        //
        // Try running your application with
        "flutter run". You'll see the
        // application has a blue toolbar. Then,
        without quitting the app, try
        // changing the primarySwatch below to
        Colors.green and then invoke
        // "hot reload" (press "r" in the console
        where you ran "flutter run",
        // or simply save your changes to "hot
        reload" in a Flutter IDE).
        // Notice that the counter didn't reset
        back to zero; the application
        // is not restarted.
        primarySwatch: Colors.blue,
      ),
      home: MyHomePage(title: 'Flutter Demo Home
Page'),
    );
  }
}

```

找到下面的代码：

```
home: MyHomePage(title: 'Flutter Demo Home  
Page'),
```

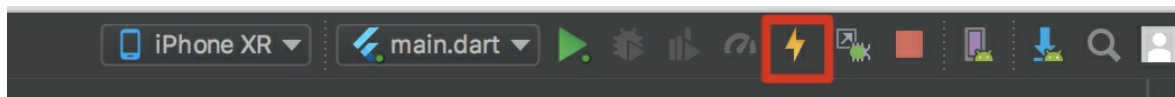
将这行代码改为：

```
home: MyHomePage(title: 'Hello World'),
```

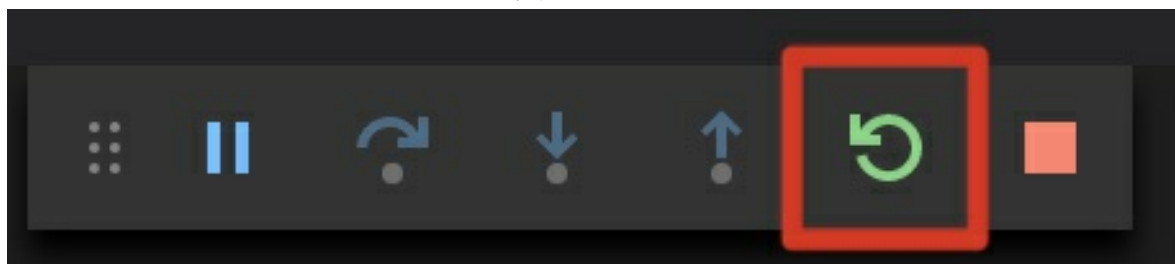
使用快捷键 **ctrl+s** (Windows、Linux) 或者 **cmd+s** (MacOS)，这个快捷键其实是保存代码，但是可以触发 Hot Reload；

或者点击 **Hot Reload** 的按钮：

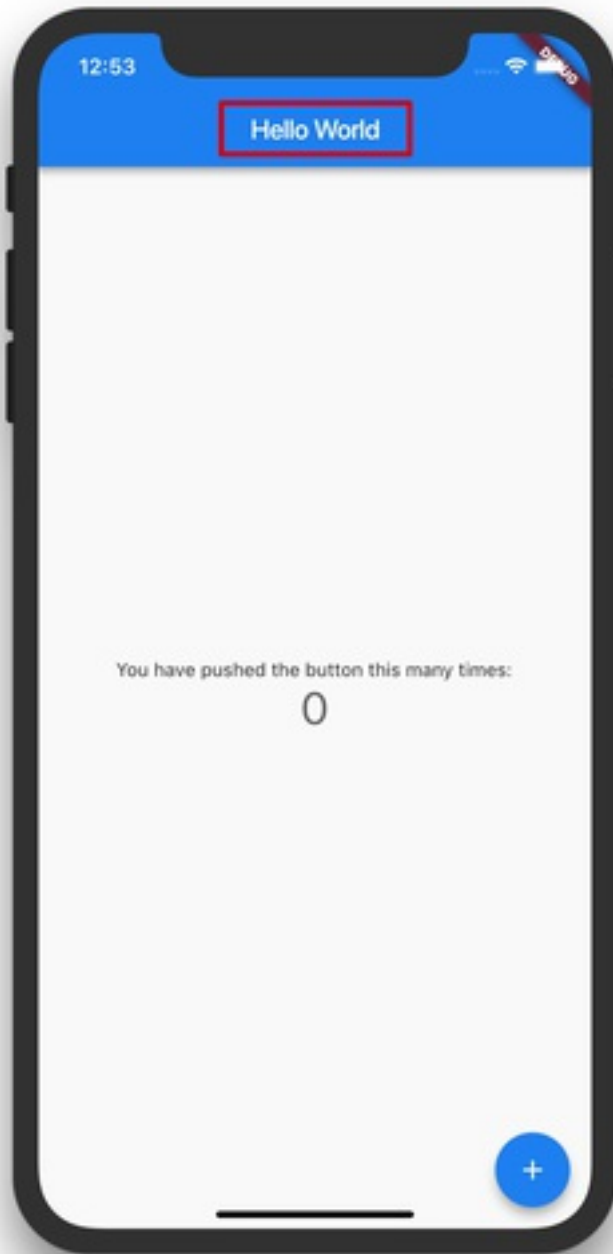
- Android Studio 的 **Hot Reload** 按钮：



- VS Code 的 **Hot Reload** 按钮：



就可以看到运行的 Flutter APP 的 TitleBar 的内容发生了变化：



是不是很神奇，不需要重新编译，就能实时看到代码的变更，这个功能就是 Flutter 的 Hot Reload。

Flutter 的 Hot Reload 功能可以帮助开发者方便快速的调试代码，构建 UI，添加功能和修复 bug。

使用 Hot Reload 的步骤

- 1.连接真机或虚拟机，运行 Flutter APP，且必须以 Debug 模式启动。因为只有 Debug 模式才能使用 Hot Reload。
- 2.修改 Flutter APP 工程里的 Dart 代码，但并不是所有 Dart 代码的修改都可以使用 Hot Reload，有一些情况下 Hot Reload 并不能生效，只能使用 Hot Restart（重新启动）。
- 3.然后使用快捷键 **ctrl+s**（Windows、Linux）或者 **cmd+s**（MacOS），或者点击 Hot Reload 的按钮。就完成了 Hot Reload 的操作，Hot Reload 成功后，会在 Debug Console 中看到输出如下类似的消息：

Reloaded 1 of 419 libraries in 1,165ms.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
Launching lib/main.dart on Android SDK built for x86 in debug mode...
Built build/app/outputs/apk/debug/app-debug.apk.
D/ (11000): HostConnection::get() New Host Connection established 0xa40a6980, tid 11023
D/EGL_emulation(11000): eglMakeCurrent: 0xa6a08340: ver 3 0 (tinfo 0xa40f9cd0)
Reloaded 1 of 432 libraries in 546ms.
```

Hot Reload 的工作原理

Hot Reload 只能在 Debug 模式下使用，是因为 Debug 模式下，Flutter 采用的是 JIT 动态编译，代码是运行在 Dart VM 上，JIT 将 Dart 编译成可以运行在 Dart VM 上的 Dart Kernel，Dart Kernel 可以动态更新，所以就实现了代码的实时更新功能。

当调用 Hot Reload 时：

1. 首先会扫描代码，找到上次编译之后有变化的 Dart 代码。
2. 在将这些变化的 Dart 代码转化为增量的 Dart Kernel 文件。
3. 将增量的 Dart Kernel 文件发送到正在移动设备上运行的 Dart VM。
4. Dart VM 会将发来的增量 Dart Kernel 文件和原有的 Dart Kernel 文件合并，然后重新加载全新的 Dart Kernel。

5. 这个时候，虽然重新加载了 Dart Kernel，**却不会重新执行代码**，而是通知 Flutter Framework 重建 Widget。

所以 Flutter 的 Hot Reload 并不会重新执行一遍代码，而是触发 Flutter 重新绘制，并且会保留 Flutter 之前的状态，所以 Hot Reload 也被称为有状态的热重载。

这里你可能会有点难以理解，什么是重建 Widget？什么是状态？没有关系，后面都会讲到，你可以继续往下看。

不能使用 Hot Reload 的场景

在理解了 Hot Reload 的原理之后，可以看到 Hot Reload 的使用场景是有一些限制的，接下来我们在看一下不能使用 Hot Reload 的场景（这里可能会有你不太理解的内容，那你可以大致看一下，有个印象，等你看完后面的内容，在来看这部分内容，就会好理解了）：

1. 代码出现编译错误的不能使用 Hot Reload

当代码更改引入编译错误时，肯定不能使用 Hot Reload。

所以要先解决编译错误，在使用 Hot Reload。

2. 代码更改会影响 APP 状态的不能使用 Hot Reload

如果你的代码更改会影响 APP 的状态，使得代码更改之后的状态和代码更改之前的状态不一样，那么 Hot Reload 就不会生效，例如：

```
class MyWidget extends StatelessWidget {  
  Widget build(BuildContext context) {  
    return GestureDetector(onTap: () =>  
print('T'));  
  }  
}
```

这段代码，运行 App 之后，将 **StatelessWidget** 改为 **StatefulWidget**：

```
class MyWidget extends StatefulWidget {  
  @override  
  State<MyWidget> createState() =>  
MyWidgetState();  
}  
  
class MyWidgetState extends State<MyWidget> {  
/* ... */ }  
}
```

因为 Hot Reload 会保留状态，在代码更改之前，MyWidget 是 **StatelessWidget**，将 **StatelessWidget** 改为 **StatefulWidget**，如果 Hot Reload 成功，那么 MyWidget 会变成 **StatefulWidget**，与它之前的状态就会不兼容的，所以 Hot Reload 是不会成功的。

3. 全局变量（global variables）和静态字段（static fields）的更改不能使用 Hot Reload

在 Flutter 中，全局变量和静态字段被视为状态，因此在 Hot Reload 期间不会重新初始化。

如下的代码：

```
final sampleTable = [  
    Table("T1"),  
    Table("T2"),  
    Table("T3"),  
    Table("T4"),  
];
```

运行 App 之后，如果做了如下的更改：

```
final sampleTable = [  
    Table("T1"),  
    Table("T2"),  
    Table("T3"),  
    Table("T10"),    // 修改这里的值  
];
```

运行 Hot Reload，是不会成功的，所以全局变量和静态字段不能使用 Hot Reload。

4. main() 方法里的更改不能使用 Hot Reload

因为 main() 方法不会因重建窗口小部件树而重新执行，所以更改 main() 方法里的代码，不会在 Hot Reload 之后看到效果。

例如，如下的代码：


```
import 'package:flutter/material.dart';

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  Widget build(BuildContext context) {
    return GestureDetector(onTap: () =>
print('tapped'));
  }
}
```

在运行App后，更改如下：

```
import 'package:flutter/widgets.dart';

void main() {
  runApp(const Center(
    child: const Text('Hello', textDirection:
TextDirection.ltr)));
}
```

Hot Reload 之后，不会看到任何变化。

5. 枚举类型更改为常规的类或者常规的类变为枚举类型也不能使用 HotReload

例如，如下的例子：

```
enum Color {  
    red,  
    green,  
    blue  
}
```

改为:

```
class Color {  
    Color(this.i, this.j);  
    final int i;  
    final int j;  
}
```

6. 修改通用类型声明也不能使用 HotReload

例如，如下的例子：

```
class A<T> {  
    T i;  
}
```

改为:

```
class A<T, V> {  
    T i;  
    V v;  
}
```

Hot Reload VS Hot Restart

针对上面不能使用 Hot Reload 的情况，就需要使用 Hot Restart。

Hot Restart 可以完全重启您的应用程序，但却不用结束调试会话。

Android Studio 里执行 Hot Restart

在 Android Studio 里，无需 stop，在 run 一下，就是 Hot Restart。

VS Code 里执行 Hot Restart

在 VS Code 里，打开命令面板，输入 `**Flutter: Hot Restart **` 或者直接快捷键 `Ctrl+F5`，就可以使用 Hot Restart。

总结

最适合 Hot Reload 的场景就是写布局的时候，如果是其他场景，尤其是写业务逻辑的时候，一不小心就会碰到无法使用 Hot Reload 的场景，所以当你发现 Hot Reload 不生效的时候，就使用 Hot Restart 吧，Hot Restart 也一样很快。