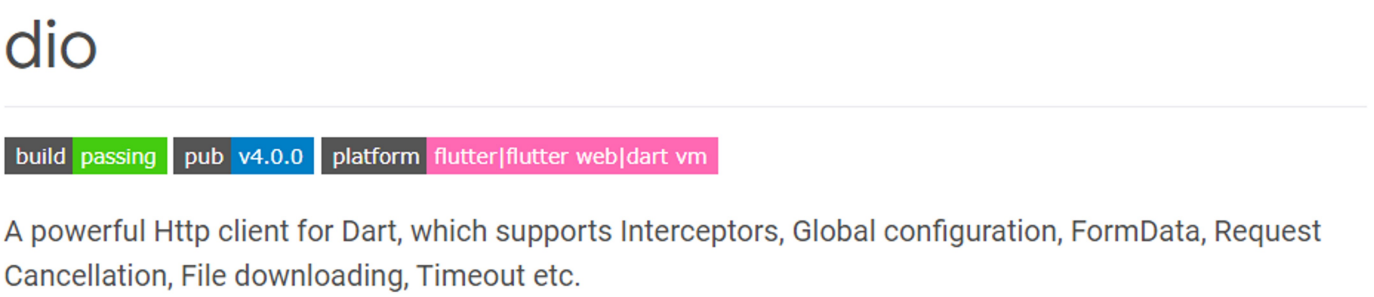
网络请求和JSON解析

2021年7月2日 8:50

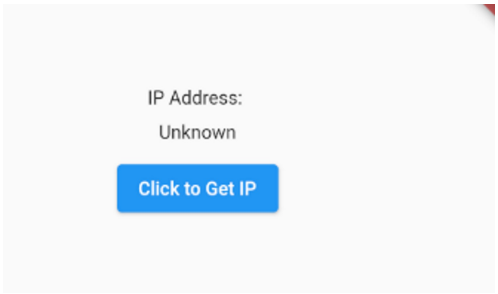
一、 Dio的使用

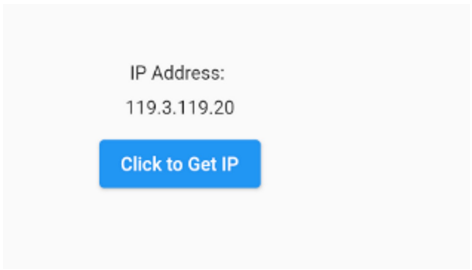
Dio是一个Dart Http请求库，支持拦截器，全局配置，请求取消，文件下载等。



更多内容可以在Dart packages (pub.dev)查看

```
classMyDioWidgetextendsStatefulWidget{
constMyDioWidget({Key?key}):super(key:key);

@override
_MyDioWidgetStatecreateState()=>_MyDioWidgetState();
}

class_MyDioWidgetStateextendsState<MyDioWidget>{

var_ipAddress='Unknown';

_getIpAddress()async{
var url='https://httpbin.org/ip';
Dio_dio=newDio();
String result;
try{
varresponse=await_dio.get(url);
if(response.statusCode==HttpStatus.ok){
vardata=jsonDecode(response.toString());
result=data['origin'];
}else{
result='ErrorgettingIPstatus${response.statusCode}';
}
}catch(exception){
result=exception.toString();
}
if(!mounted)return;
setState((){
_ipAddress=result;
});
}
```





Dart中也是通过async和await关键字来支持异步编程的，async使得方法变成异步方法
await关键字只能在异步方法中使用，它使得之后的代码等到get请求返回后再执行。

二、JSON数据解析
常用的JSON数据解析方式主要有以下三种。
2.1 使用json.decode()方法
json.decode()方法会将String类型数据解析成Map对象Map<String, dynamic>

2.2 手动编写实体类
即把代码封装。

```
classIp{
String origin;
Ip(this.origin);

Ip.fromJson(Map<String,dynamic>json):origin=json['origin'];

Map<String,dynamic> toJson()=>{
"origin":origin
};
}
```

2.3 使用第三方自动生成工具
比如 Instantly parse JSON in any language | quicktype
以及 json_serializable