

Conectando tu app Flutter con el mundo exterior

Dio como puente entre APIs y Flutter

Contenido

01

Fundamentos

02

Dio

03


Dart 3



01

Fundamentos

Teoría y más teoría





Programación síncrona vs asíncrona





DartPad

<> New Pad

↺ Reset

≡ Format

⬇ Install SDK

silent-tulip-3886

```
1 ▼ void main() {  
2   final p = Persona();  
3   p.caminar();  
4   p.caminar();  
5   p.caminar();  
6   p.darCharla();  
7 }  
8  
9 ► class Persona { ... }  
1
```

▶ Run

Console

Mover pierna derecha
Mover pierna izquierda
Mover pierna derecha
Hola :)

```
void main() {
    final p = Persona();
    p.caminar();

    print('\tLanzamiento inicial');
    p.lanzarPelota().then((volvio) {
        print(volvio ? '\tPuedes lanzar otra vez' : '\tPerdiste la pelota');
    }).catchError((_) {
        print('\tQue pelota? yo no vi nada');
    });

    p.caminar();
    p.darCharla();
    p.caminar();
}

class Persona {
    var move = true;

    Future<bool> lanzarPelota() {
        return Future.delayed(const Duration(seconds: 3), () => true);
    }

    void caminar() {...}

    void darCharla() {...}
}
```

▶ Run

Console

```
Mover pierna derecha
      Lanzamiento inicial
Mover pierna izquierda
Hola :)
Mover pierna derecha
      Puedes lanzar otra vez
```


Documentation

```
1 ▼ void main() async {
2   final p = Persona();
3   p.caminar();
4
5   print('Esperando camara...');
6   final camStatus = await p.alistarCamara();
7 ▼ if (camStatus) {
8   print('Todos digan whiskyyy');
9   final img = await p.tomarFoto();
0   print('Imagen $img');
1 }
2
3   p.darCharla();
4 }
5
6 ▶ class Persona { ... }
0
```


▶ Run

Console

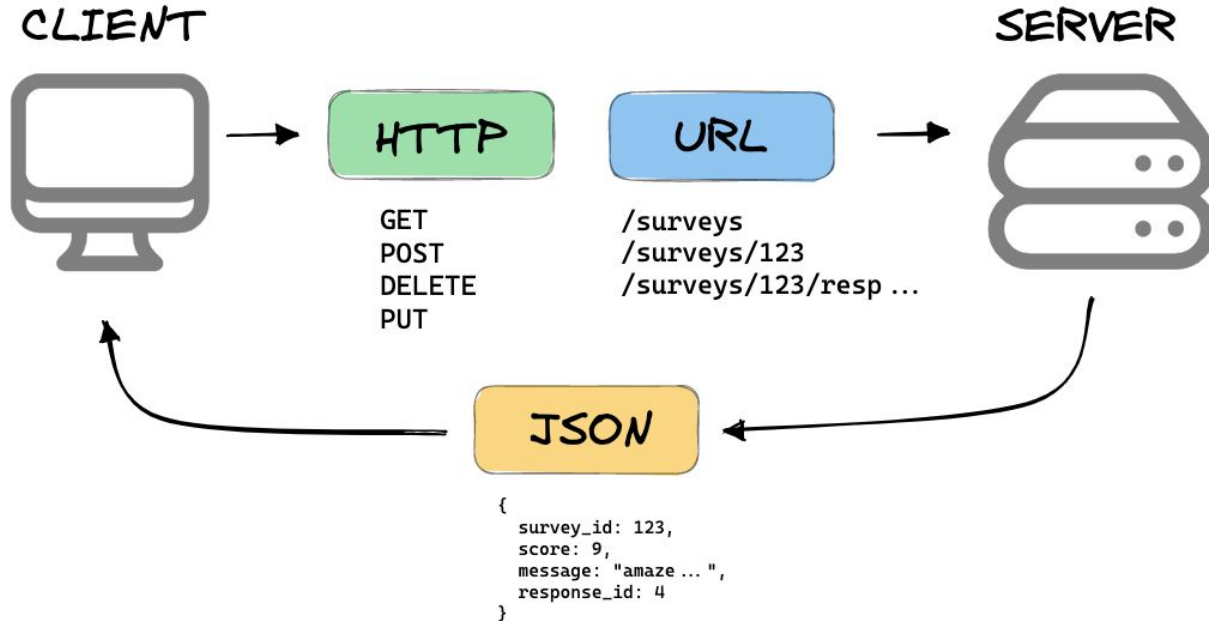
Mover pierna derecha
Esperando camara...
Todos digan whiskyyy
Imagen IMG-001.jpg
Hola :)



Application Programming Interface



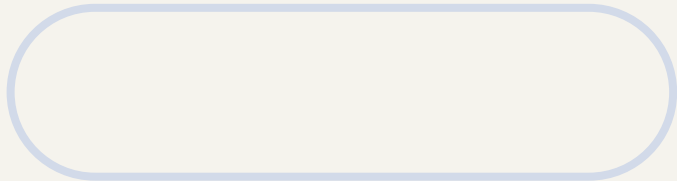
WHAT IS A REST API?





<https://reqres.in>

<https://reqres.in/api-docs/#/>





02

Dio



<https://pub.dev/packages/dio>

Beneficios

Manejo de Errores

Dio cuenta con un sistema de manejo de errores más granular.

Interceptores

Permite intervenir en las solicitudes y respuestas HTTP.

Documentación y Comunidad

Documentación completa y una comunidad activa.

GET



```
1 Future<dynamic> _getExample() async {  
2   final response = await Dio().get('https://reqres.in/api/users/2');  
3   return response.data;  
4 }
```

POST



```
1 Future<String> _postExample(Map<String, dynamic> data) async {  
2   try {  
3     final response = await Dio().post(  
4       'https://reqres.in/api/login',  
5       data: data,  
6     );  
7     return '${response.data}';  
8   } on DioException catch (e) {  
9     return 'Error dio $e';  
10  }  
11 }
```

PUT/PATCH



```
1 Future<String> _patchExample(String userId, Map<String, dynamic> data) async {  
2   try {  
3     final response = await Dio().patch(  
4       'https://reqres.in/api/users/$userId',  
5       data: data,  
6     );  
7     return '${response.data}';  
8   } on DioException catch (e) {  
9     return 'Error dio $e';  
10  }  
11 }
```

DELETE



```
1  Future<String> _deleteExample(String userId) async {  
2      try {  
3          final response = await Dio().delete(  
4              'https://reqres.in/api/users/$userId',  
5          );  
6          return '${response.statusCode}';  
7      } on DioException catch (e) {  
8          return 'Error dio $e';  
9      }  
10 }
```


¿Qué podemos mejorar
de este código?



```
1 DioClientException(DioException dioError) {  
2   switch (dioError.type) {  
3     case DioExceptionType.connectionTimeout:  
4       errorMessage = 'Se ha agotado el tiempo de conexión.';  
5       break;  
6     case DioExceptionType.badResponse:  
7       errorMessage = _errorStatusCode(dioError.response?.statusCode);  
8       break;  
9     case DioExceptionType.unknown:  
10      if (dioError.error is SocketException) {  
11        errorMessage = 'No tienes conexión a internet.';  
12        break;  
13      }  
14      errorMessage = 'Ocurrió un error inesperado';  
15      break;  
16    default:  
17      errorMessage = 'Algo salió mal!.';  
18      break;  
19  }  
20 }
```



```
1  String _errorStatusCode(int? statusCode) {  
2      switch (statusCode) {  
3          case 404:  
4              return 'Elemento no encontrado.';  
5          case 500:  
6              return 'Internal server error.';  
7          default:  
8              return 'Algo salio mal!';  
9      }  
10 }
```



```
1  class DioClient {
2      DioClient._();
3
4      static final instance = DioClient._();
5
6      final _dio = Dio(
7          BaseOptions(
8              baseUrl: 'https://reqres.in/api',
9              connectTimeout: const Duration(seconds: 3),
10          ),
11      );
12
13      bool _isSuccess(int? code) {
14          return code != null && code >= 200 && code <= 299;
15      }
16
17      Future<Map<String, dynamic>> get({...}) {}
18
19      Future<Map<String, dynamic>> post({...}) {}
20
21 }
```



```
1  Future<Map<String, dynamic>> get({
2    required String path,
3    Options? options,
4    Map<String, dynamic>? queryParams,
5  }) async {
6    try {
7      final response = await _dio.get(
8        path,
9        options: options,
10       queryParameters: queryParams,
11     );
12
13     if (_isSuccess(response.statusCode)) {
14       return response.data;
15     }
16     throw 'Error get $path';
17   } on DioException catch (dioError) {
18     final error = DioClientException(dioError);
19     throw error.errorMessage;
20   }
21 }
```



```
1 Future<dynamic> _getExample() async {  
2   final response = await Dio().get('https://reqres.in/api/users/2');  
3   return response.data;  
4 }  
5  
6 Future<dynamic> _getExample2() async {  
7   final data = await DioClient.instance.get(  
8     path: '/users/3',  
9   );  
10  return data;  
11 }
```



```
1  body: FutureBuilder(  
2    future: error ? _getExampleError() : _getExample(),  
3    builder: (context, snapshot) {  
4      if (snapshot.hasData) {  
5        return Center(  
6          child: Text(  
7            '${snapshot.data}',  
8            style: textTheme.bodyLarge,  
9          ),  
10       );  
11     }  
12  
13     if (snapshot.hasError) {  
14       return Center(  
15         child: Text(  
16           '${snapshot.error}',  
17           style: textTheme.bodyLarge,  
18         ),  
19       );  
20     }  
21     return const Center(child: CircularProgressIndicator());  
22   },  
23 ),
```

Interceptors



```
1 class DioClient {
2   DioClient._();
3
4   static final instance = DioClient._();
5
6   final _dio = Dio(
7     BaseOptions(
8       baseUrl: 'https://reqres.in/api',
9       connectTimeout: const Duration(seconds: 3),
10    ),
11  )..interceptors.add(LogInterceptor(
12    request: true,
13    requestHeader: false,
14    responseBody: true,
15    responseHeader: false,
16  ));
```


Interceptors

```
I/flutter (24215): *** Request ***
I/flutter (24215): uri: https://reqres.in/api/users/3
I/flutter (24215): method: GET
I/flutter (24215): responseType: ResponseType.json
I/flutter (24215): followRedirects: true
I/flutter (24215): persistentConnection: true
I/flutter (24215): connectTimeout: 0:00:03.000000
I/flutter (24215): sendTimeout: null
I/flutter (24215): receiveTimeout: null
I/flutter (24215): receiveDataWhenStatusError: true
I/flutter (24215): extra: {}
I/flutter (24215):
I/flutter (24215): *** Response ***
I/flutter (24215): uri: https://reqres.in/api/users/3
I/flutter (24215): Response Text:
I/flutter (24215): {"data":{"id":3,"email":"emma.wong@reqres.in","first_name":"Emma","last_name":"Wong","avatar":"https://reqres.in/
rt":{"url":"https://reqres.in/#support-heading","text":"To keep ReqRes free, contributions towards server costs are appreciated!"}}
```

Interceptors

```
1  final _dio = Dio(  
2    BaseOptions(  
3      baseUrl: 'https://reqres.in/api',  
4      connectTimeout: const Duration(seconds: 3),  
5    ),  
6  )..interceptors.addAll(  
7    [  
8      InterceptorsWrapper(  
9        onRequest: (options, handler) async {  
10          if (options.path != '/login' && options.path != '/register') {  
11            const token = 'El token viene aqui';  
12            options.headers.addAll({'Authorization': 'Bearer $token'});  
13          }  
14          handler.next(options);  
15        },  
16      ),  
17      if (kDebugMode)  
18        LogInterceptor(  
19          request: true,  
20          requestHeader: true,  
21          responseBody: true,  
22          responseHeader: false,  
23        ),  
24    ],  
25  );
```

Interceptors

```
I/flutter (10903): *** Request ***  
I/flutter (10903): uri: https://reqres.in/api/users/3  
I/flutter (10903): method: GET  
I/flutter (10903): responseType: ResponseType.json  
I/flutter (10903): followRedirects: true  
I/flutter (10903): persistentConnection: true  
I/flutter (10903): connectTimeout: 0:00:03.000000  
I/flutter (10903): sendTimeout: null  
I/flutter (10903): receiveTimeout: null  
I/flutter (10903): receiveDataWhenStatusError: true  
I/flutter (10903): extra: {}  
I/flutter (10903): headers:  
I/flutter (10903):   Authorization: Bearer El token viene aqui  
I/flutter (10903):  
I/flutter (10903): *** Response ***  
I/flutter (10903): uri: https://reqres.in/api/users/3  
I/flutter (10903): Response Text:
```

Interceptors

```
I/flutter (10903): *** Request ***  
I/flutter (10903): uri: https://reqres.in/api/login  
I/flutter (10903): method: POST  
I/flutter (10903): responseType: ResponseType.json  
I/flutter (10903): followRedirects: true  
I/flutter (10903): persistentConnection: true  
I/flutter (10903): connectTimeout: 0:00:03.000000  
I/flutter (10903): sendTimeout: null  
I/flutter (10903): receiveTimeout: null  
I/flutter (10903): receiveDataWhenStatusError: true  
I/flutter (10903): extra: {}  
I/flutter (10903): headers:  
I/flutter (10903):  content-type: application/json  
I/flutter (10903):  
I/flutter (10903): *** DioException ***:  
I/flutter (10903): uri: https://reqres.in/api/login
```



03

Dart 3

<https://dart.dev>



Novedades

100% sound null safety

Record, patterns, and class modifiers

Compilation to WebAssembly



```
1  (String, int) userInfo() {  
2      return ('Juan', 23);  
3  }  
4  
5  void main() {  
6      final data = userInfo();  
7      data.$1; // Juan  
8      data.$2; // 23  
9  }
```



```
1 String message(int day) {  
2     return switch (day) {  
3         >= 1 || <= 5 => 'Trabajo muy duro como un esclavo',  
4         == 6 || == 7 => 'Wiiiiii!!!',  
5         _ => 'Algo salio mal',  
6     };  
7 }
```




```
1  sealed class Animal { ... }  
2  
3  class Cow extends Animal { ... }  
4  class Sheep extends Animal { ... }  
5  class Pig extends Animal { ... }
```



```
1 sealed class ErrorRequest {}
2
3 class NetworkError extends ErrorRequest {}
4
5 class TimeoutError extends ErrorRequest {
6     final String path;
7
8     TimeoutError({required this.path});
9 }
10
11 class ServerError extends ErrorRequest {
12     final int statusCode;
13     final String path;
14     final dynamic cause;
15
16     ServerError({required this.statusCode, required this.path, required this.cause});
17 }
```



```
1 DioClientException(DioException dioError) {
2   switch (dioError.type) {
3     case DioExceptionType.connectionTimeout:
4       // error = 'Se ha agotado el tiempo de conexión.';
5       error = TimeoutError(path: dioError.requestOptions.path);
6       break;
7     case DioExceptionType.badResponse:
8       error = _errorStatusCode(dioError.response?.statusCode);
9       break;
10    case DioExceptionType.unknown:
11      if (dioError.error is SocketException) {
12        // error = 'No tienes conexión a internet.';
13        error = NetworkError();
14        break;
15      }
16      error = ServerError(
17        statusCode: dioError.response?.statusCode,
18        path: dioError.requestOptions.path,
19        cause: dioError.message,
20      );
21      break;
22    default:
23      error = ServerError(
24        statusCode: dioError.response?.statusCode,
25        path: dioError.requestOptions.path,
26        cause: dioError.message,
27      );
28      break;
29  }
30 }
```



```
1 Future<Map<String, dynamic>> get({
2   required String path,
3   Options? options,
4   Map<String, dynamic>? queryParams,
5 }) async {
6   try {
7     final response = await _dio.get(
8       path,
9       options: options,
10      queryParameters: queryParams,
11    );
12
13    if (_isSuccess(response.statusCode)) {
14      return response.data;
15    }
16    throw 'Error get $path';
17  } on DioException catch (dioError) {
18    throw DioClientException(dioError).error;
19  }
20 }
```

```
1  body: FutureBuilder(  
2    future: error ? _getExampleError() : _getExample(),  
3    builder: (context, snapshot) {  
4      if (snapshot.hasData) {  
5        return Center(  
6          child: Text(  
7            '${snapshot.data}',  
8            style: textTheme.bodyLarge,  
9          ),  
10       );  
11     }  
12  
13     if (snapshot.hasError) {  
14       final error = snapshot.error as ErrorRequest;  
15       return switch (error) {  
16         NetworkError() => const SizedBox(),  
17         ServerError() => const SizedBox(),  
18         TimeoutError(path: final path) => Center(  
19           child: Text(  
20             'Timeout $path',  
21             style: textTheme.bodyLarge,  
22           ),  
23         ),  
24       };  
25     }  
26     return const Center(child: CircularProgressIndicator());  
27   },  
28 ),
```

Gracias!



Adalid O. Limachi L.



@eldetulado



69843755

