

ZHttp Direct Suspended Asynchronous

```
1 val client = ZHttpClient.Builder()
2   .baseUrl(BASE_URL)
3   .connectionTimeout(CONNECTION_TIME_OUT) // optional
4   .readTimeout(READ_TIME_OUT) // optional
5   .defaultHeaders(DEFAULT_HEADERS) // optional
6   .filesBufferSize(BUFFER_SIZE) // optional
7   .build()
8
9 // The syntax of a GET request.
10 val response = client.get<TYPE>(ENDPOINT, QUERIES, HEADERS)
11
12 // The syntax of a POST request.
13 val response = client.post<TYPE>(ENDPOINT, BODY, QUERIES, HEADERS)
14
15 // The syntax of a DELETE request.
16 val response = client.delete<TYPE>(ENDPOINT, QUERIES, HEADERS)
17
18 // The syntax of a PUT request.
19 val response = client.put<TYPE>(ENDPOINT, BODY, QUERIES, HEADERS)
20
21 // The syntax of a PATCH request.
22 val response = client.patch<TYPE>(ENDPOINT, BODY, QUERIES, HEADERS)
23
24 // The syntax of a MULTIPART request.
25 val response = client.multiPart<TYPE>(ENDPOINT, parts, QUERIES, HEADERS)
26
```

ZHttp Asynchronous Lambda

```
1 val client = ZHttpClient.Builder()
2   .baseUrl(BASE_URL)
3   .connectionTimeout(CONNECTION_TIME_OUT) // optional
4   .readTimeout(READ_TIME_OUT) // optional
5   .defaultHeaders(DEFAULT_HEADERS) // optional
6   .filesBufferSize(BUFFER_SIZE) // optional
7   .build()
8
9 client.get<TYPE>(ENDPOINT, QUERIES, HEADERS) { success, failure ->
10   // Handle Success or Failure.
11 }
12
13 client.post<TYPE>(ENDPOINT, BODY, QUERIES, HEADERS) { success, failure ->
14   // Handle Success or Failure.
15 }
16
17 client.delete<TYPE>(ENDPOINT, QUERIES, HEADERS) { success, failure ->
18   // Handle Success or Failure.
19 }
20
21 client.put<TYPE>(ENDPOINT, BODY, QUERIES, HEADERS) { success, failure ->
22   // Handle Success or Failure.
23 }
24
25 client.patch<TYPE>(ENDPOINT, BODY, QUERIES, HEADERS) { success, failure ->
26   // Handle Success or Failure.
27 }
28
29 client.multiPart<TYPE>(ENDPOINT, PARTS, QUERIES, HEADERS) { success, failure ->
30   // Handle Success or Failure.
31 }
32
```

ZHttp Raw Synchronous

```
1 val client = ZHttpClient.Builder()
2   .baseUrl(BASE_URL)
3   .connectionTimeout(CONNECTION_TIME_OUT) // optional
4   .readTimeout(READ_TIME_OUT) // optional
5   .defaultHeaders(DEFAULT_HEADERS) // optional
6   .filesBufferSize(BUFFER_SIZE) // optional
7   .build()
8
9 // The syntax of a synchronous GET request.
10 val response = ZGet(client).doGet(ENDPOINT, QUERIES, HEADERS)
11
12 // The syntax of a synchronous POST request.
13 val response = ZPost(client).doPost(ENDPOINT, BODY, QUERIES, HEADERS)
14
15 // The syntax of a synchronous DELETE request.
16 val response = ZDelete(client).doDelete(ENDPOINT, QUERIES, HEADERS)
17
18 // The syntax of a synchronous PUT request.
19 val response = ZPut(client).doPut(ENDPOINT, BODY, QUERIES, HEADERS)
20
21 // The syntax of a synchronous PATCH request.
22 val response = ZPatch(client).doPatch(ENDPOINT, BODY, QUERIES, HEADERS)
23
24 // The syntax of a synchronous MULTIPART request.
25 val response = ZMultipart(client).doMultipart(ENDPOINT, PARTS, QUERIES, HEADERS)
26
```