ZHttp Direct Suspended Asynchronous

```
val client = ZHttpClient.Builder()
    .baseUrl(BASE_URL)
    .connectionTimeout(CONNECTION_TIME_OUT) // optional
    .readTimeout(READ_TIME_OUT) // optional
    .defaultHeaders(DEFAULT_HEADERS) // optional
    .filesBufferSize(BUFFER_SIZE) // optional
    .build()

// The syntax of a GET request.
val response = client.get<TYPE>(ENDPOINT, QUERIES, HEADERS)

// The syntax of a POST request.
val response = client.post<TYPE>(ENDPOINT, BODY, QUERIES, HEADERS)

// The syntax of a DELETE request.
val response = client.delete<TYPE>(ENDPOINT, QUERIES, HEADERS)

// The syntax of a PUT request.
val response = client.put<TYPE>(ENDPOINT, BODY, QUERIES, HEADERS)

// The syntax of a PATCH request.
val response = client.patch<TYPE>(ENDPOINT, BODY, QUERIES, HEADERS)

// The syntax of a MULTIPART request.
val response = client.multiPart<TYPE>(ENDPOINT, parts, QUERIES, HEADERS)
```

ZHttp Asynchronous Lambda

```
val client = ZHttpClient.Builder()
    .baseUrl(BASE_URL)
    .connectionTimeout(CONNECTION_TIME_OUT) // optional
    .readTimeout(READ_TIME_OUT) // optional
    .defaultHeaders(DEFAULT_HEADERS) // optional
    .filesBufferSize(BUFFER_SIZE) // optional
    .build()

client.get<TYPE>(ENDPOINT, QUERIES, HEADERS) { success, failure ->
    // Handle Success or Failure.
}

client.post<TYPE>(ENDPOINT, BODY, QUERIES, HEADERS) { success, failure ->
    // Handle Success or Failure.
}

client.delete<TYPE>(ENDPOINT, QUERIES, HEADERS) { success, failure ->
    // Handle Success or Failure.
}

client.put<TYPE>(ENDPOINT, BODY, QUERIES, HEADERS) { success, failure ->
    // Handle Success or Failure.
}

client.patch<TYPE>(ENDPOINT, BODY, QUERIES, HEADERS) { success, failure ->
    // Handle Success or Failure.
}

client.multiPart<TYPE>(ENDPOINT, PARTS, QUERIES, HEADERS) { success, failure ->
    // Handle Success or Failure.
}
```

ZHttp Raw Synchronous

```
val client = ZHttpClient.Builder()
    .baseUrl(BASE_URL)
    .connectionTimeout(CONNECTION_TIME_OUT) // optional
    .readTimeout(READ_TIME_OUT) // optional
    .defaultHeaders(DEFAULT_HEADERS) // optional
    .filesBufferSize(BUFFER_SIZE) // optional
    .build()

// The syntax of a synchronous GET request.
val response = ZGet(client).doGet(ENDPOINT, QUERIES, HEADERS)

// The syntax of a synchronous POST request.
val response = ZPost(client).doPost(ENDPOINT, BODY, QUERIES, HEADERS)

// The syntax of a synchronous DELETE request.
val response = ZDelete(client).doDelete(ENDPOINT, QUERIES, HEADERS)

// The syntax of a synchronous PUT request.
val response = ZPut(client).doPut(ENDPOINT, BODY, QUERIES, HEADERS)

// The syntax of a synchronous PATCH request.
val response = ZPatch(client).doPatch(ENDPOINT, BODY, QUERIES, HEADERS)

// The syntax of a synchronous MULTIPART request.
val response = ZMultipart(client).doMultipart(ENDPOINT, PARTS, QUERIES, HEADERS)
```