Java REST Client

Unresolved directive in ../Versions.asciidoc - include::{asciidoc-dir}/../../shared/attributes.asciidoc[]

Overview

The Java REST Client comes in 2 flavors:

- Java Low Level REST Client: the official low-level client for Elasticsearch. It allows to communicate with an Elasticsearch cluster through http. Leaves requests marshalling and responses un-marshalling to users. It is compatible with all Elasticsearch versions.
- Java High Level REST Client: the official high-level client for Elasticsearch. Based on the low-level client, it exposes API specific methods and takes care of requests marshalling and responses un-marshalling.

Java Low Level REST Client

The low-level client's features include:

- minimal dependencies
- load balancing across all available nodes
- failover in case of node failures and upon specific response codes
- failed connection penalization (whether a failed node is retried depends on how many consecutive times it failed; the more failed attempts the longer the client will wait before trying that same node again)
- persistent connections
- trace logging of requests and responses
- optional automatic discovery of cluster nodes

Getting started

This section describes how to get started with the low-level REST client from getting the artifact to using it in an application.

Javadoc

The javadoc for the low level REST client can be found at https://artifacts.elastic.co/javadoc/org/elasticsearch/client/elasticsearch-rest-client/6.1.1/index.html.

Maven Repository

The low-level Java REST client is hosted on Maven Central. The minimum Java version required is `1.7`.

The low-level REST client is subject to the same release cycle as Elasticsearch. Replace the version with the desired client version, first released with `5.0.0-alpha4`. There is no relation between the client version and the Elasticsearch version that the client can communicate with. The low-level REST client is compatible with all Elasticsearch versions.

Maven configuration

Here is how you can configure the dependency using maven as a dependency manager. Add the following to your `pom.xml` file:

```
<dependency>
    <groupId>org.elasticsearch.client</groupId>
    <artifactId>elasticsearch-rest-client</artifactId>
    <version>6.1.1</version>
</dependency>
```

Gradle configuration

Here is how you can configure the dependency using gradle as a dependency manager. Add the following to your `build.gradle` file:

```
dependencies {
    compile 'org.elasticsearch.client:elasticsearch-rest-client:6.1.1'
}
```

Dependencies

The low-level Java REST client internally uses the Apache Http Async Client to send http requests. It depends on the following artifacts, namely the async http client and its own transitive dependencies:

- org.apache.httpcomponents:httpasyncclient
- org.apache.httpcomponents:httpcore-nio
- org.apache.httpcomponents:httpclient
- org.apache.httpcomponents:httpcore
- commons-codec:commons-codec
- commons-logging:commons-logging

Shading

In order to avoid version conflicts, the dependencies can be shaded and packaged within the client in a single JAR file (sometimes called an "uber JAR" or "fat JAR"). Shading a dependency consists of taking its content (resources files and Java class files) and renaming some of its packages before putting them in the same JAR file as the low-level Java REST client. Shading a JAR can be accomplished by 3rd-party plugins for Gradle and Maven.

Be advised that shading a JAR also has implications. Shading the Commons Logging layer, for instance, means that 3rd-party logging backends need to be shaded as well.

Maven configuration

Here is a configuration using the Maven Shade plugin. Add the following to your `pom.xml` file:

```
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-shade-plugin</artifactId>
            <version>3.1.0</version>
            <executions>
                <execution>
                    <phase>package</phase>
                    <goals><goal>shade</goal></goals>
                    <configuration>
                        <relocations>
                            <relocation>
                                <pattern>org.apache.http</pattern>
                                <shadedPattern>hidden.org.apache.http</shadedPattern>
                            </relocation>
                            <relocation>
                                <pattern>org.apache.logging</pattern>
                                <shadedPattern>
hidden.org.apache.logging</shadedPattern>
                            </relocation>
                            <relocation>
                                <pattern>org.apache.commons.codec</pattern>
                                <shadedPattern>
hidden.org.apache.commons.codec</shadedPattern>
                            </relocation>
                            <relocation>
                                <pattern>org.apache.commons.logging</pattern>
                                <shadedPattern>
hidden.org.apache.commons.logging</shadedPattern>
                            </relocation>
                        </relocations>
                    </configuration>
                </execution>
            </executions>
        </plugin>
    </plugins>
</build>
```

Gradle configuration

Here is a configuration using the Gradle ShadowJar plugin. Add the following to your `build.gradle` file:

```
shadowJar {
    relocate 'org.apache.http', 'hidden.org.apache.http'
    relocate 'org.apache.logging', 'hidden.org.apache.logging'
    relocate 'org.apache.commons.codec', 'hidden.org.apache.commons.codec'
    relocate 'org.apache.commons.logging', 'hidden.org.apache.commons.logging'
}
```

Initialization

A `RestClient` instance can be built through the corresponding `RestClientBuilder` class, created via `RestClient#builder(HttpHost…)` static method. The only required argument is one or more hosts that the client will communicate with, provided as instances of HttpHost as follows:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-init]
```

The `RestClient` class is thread-safe and ideally has the same lifecycle as the application that uses it. It is important that it gets closed when no longer needed so that all the resources used by it get properly released, as well as the underlying http client instance and its threads:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-close]
```

`RestClientBuilder` also allows to optionally set the following configuration parameters while building the `RestClient` instance:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-init-default-headers]
```

① Set the default headers that need to be sent with each request, to prevent having to specify them with each single request

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-init-max-retry-timeout]
```

① Set the timeout that should be honoured in case multiple attempts are made for the same request. The default value is 30 seconds, same as the default socket timeout. In case the socket timeout is customized, the maximum retry timeout should be adjusted accordingly

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-init-failure-listener]
```

① Set a listener that gets notified every time a node fails, in case actions need to be taken. Used internally when sniffing on failure is enabled.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-init-request-config-callback]
```

① Set a callback that allows to modify the default request configuration (e.g. request timeouts, authentication, or anything that the `org.apache.http.client.config.RequestConfig.Builder` allows to set)

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-init-client-config-callback]
```

① Set a callback that allows to modify the http client configuration (e.g. encrypted communication over ssl, or anything that the `org.apache.http.impl.nio.client.HttpAsyncClientBuilder` allows to set)

Performing requests

Once the `RestClient` has been created, requests can be sent by calling one of the available `performRequest` or `performRequestAsync` method variants. The `performRequest` methods are synchronous and return the `Response` directly, meaning that the client will block and wait for a response to be returned. The `performRequestAsync` variants return `void` and accept an extra `ResponseListener` as an argument instead, meaning that they are executed asynchronously. The provided listener will be notified upon request completion or failure.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-verb-endpoint]
```

① Send a request by providing only the verb and the endpoint, minimum set of required arguments

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-verb-endpoint-params]
```

① Send a request by providing the verb, the endpoint, and some querystring parameter

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-verb-endpoint-params-body]
```

① Send a request by providing the verb, the endpoint, optional querystring parameters and the request body enclosed in an `org.apache.http.HttpEntity` object

IMPORTANT	The `ContentType` specified for the `HttpEntity` is important because it will be used to set the `Content-Type` header so that Elasticsearch can properly parse the content.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-response-consumer]
```

① Send a request by providing the verb, the endpoint, optional querystring parameters, optional request body and the optional factory that is used to create an `org.apache.http.nio.protocol.HttpAsyncResponseConsumer` callback instance per request attempt. Controls how the response body gets streamed from a non-blocking HTTP connection on the client side. When not provided, the default implementation is used which buffers the whole response body in heap memory, up to 100 MB.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-verb-endpoint-async]
```

① Define what needs to happen when the request is successfully performed

② Define what needs to happen when the request fails, meaning whenever there's a connection error or a response with error status code is returned.

③ Send an async request by providing only the verb, the endpoint, and the response listener to be notified once the request is completed, minimum set of required arguments

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-verb-endpoint-params-async]
```

① Send an async request by providing the verb, the endpoint, some querystring parameter and the response listener to be notified once the request is completed

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-verb-endpoint-params-body-async]
```

① Send an async request by providing the verb, the endpoint, optional querystring parameters, the request body enclosed in an `org.apache.http.HttpEntity` object and the response listener to be

notified once the request is completed

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-response-consumer-async]
```

① Send an async request by providing the verb, the endpoint, optional querystring parameters, optional request body and the optional factory that is used to create an `org.apache.http.nio.protocol.HttpAsyncResponseConsumer` callback instance per request attempt. Controls how the response body gets streamed from a non-blocking HTTP connection on the client side. When not provided, the default implementation is used which buffers the whole response body in heap memory, up to 100 MB.

The following is a basic example of how async requests can be sent:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-async-example]
```

① Process the returned response

② Handle the returned exception, due to communication error or a response with status code that indicates an error

Each of the above listed method supports sending headers along with the request through a `Header` varargs argument as in the following examples:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-headers]
```

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-headers-async]
```

Reading responses

The `Response` object, either returned by the synchronous `performRequest` methods or received as an argument in `ResponseListener#onSuccess(Response)`, wraps the response object returned by the http client and exposes some additional information.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-response2]
```

① Information about the performed request

② The host that returned the response

③ The response status line, from which you can for instance retrieve the status code

④ The response headers, which can also be retrieved by name though `getHeader(String)`

⑤ The response body enclosed in an `org.apache.http.HttpEntity` object

When performing a request, an exception is thrown (or received as an argument in `ResponseListener#onFailure(Exception)` in the following scenarios:

`IOException`

communication problem (e.g. SocketTimeoutException)

`ResponseException`

a response was returned, but its status code indicated an error (not `2xx`). A `ResponseException` originates from a valid http response, hence it exposes its corresponding `Response` object which gives access to the returned response.

NOTE

A `ResponseException` is **not** thrown for `HEAD` requests that return a `404` status code because it is an expected `HEAD` response that simply denotes that the resource is not found. All other HTTP methods (e.g., `GET`) throw a `ResponseException` for `404` responses unless the `ignore` parameter contains `404`. `ignore` is a special client parameter that doesn't get sent to Elasticsearch and contains a comma separated list of error status codes. It allows to control whether some error status code should be treated as an expected response rather than as an exception. This is useful for instance with the get api as it can return `404` when the document is missing, in which case the response body will not contain an error but rather the usual get api response, just without the document as it was not found.

Note that the low-level client doesn't expose any helper for json marshalling and un-marshalling. Users are free to use the library that they prefer for that purpose.

The underlying Apache Async Http Client ships with different `org.apache.http.HttpEntity` implementations that allow to provide the request body in different formats (stream, byte array, string etc.). As for reading the response body, the `HttpEntity#getContent` method comes handy which returns an `InputStream` reading from the previously buffered response body. As an alternative, it is possible to provide a custom `org.apache.http.nio.protocol.HttpAsyncResponseConsumer` that controls how bytes are read and buffered.

Logging

The Java REST client uses the same logging library that the Apache Async Http Client uses: Apache Commons Logging, which comes with support for a number of popular logging implementations. The java packages to enable logging for are `org.elasticsearch.client` for the client itself and `org.elasticsearch.client.sniffer` for the sniffer.

The request tracer logging can also be enabled to log every request and corresponding response in

curl format. That comes handy when debugging, for instance in case a request needs to be manually executed to check whether it still yields the same response as it did. Enable trace logging for the `tracer` package to have such log lines printed out. Do note that this type of logging is expensive and should not be enabled at all times in production environments, but rather temporarily used only when needed.

Common configuration

As explained in Initialization, the `RestClientBuilder` supports providing both a `RequestConfigCallback` and an `HttpClientConfigCallback` which allow for any customization that the Apache Async Http Client exposes. Those callbacks make it possible to modify some specific behaviour of the client without overriding every other default configuration that the `RestClient` is initialized with. This section describes some common scenarios that require additional configuration for the low-level Java REST Client.

Timeouts

Configuring requests timeouts can be done by providing an instance of `RequestConfigCallback` while building the `RestClient` through its builder. The interface has one method that receives an instance of `org.apache.http.client.config.RequestConfig.Builder` as an argument and has the same return type. The request config builder can be modified and then returned. In the following example we increase the connect timeout (defaults to 1 second) and the socket timeout (defaults to 30 seconds). Also we adjust the max retry timeout accordingly (defaults to 30 seconds too).

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-config-timeouts]
```

Number of threads

The Apache Http Async Client starts by default one dispatcher thread, and a number of worker threads used by the connection manager, as many as the number of locally detected processors (depending on what `Runtime.getRuntime().availableProcessors()` returns). The number of threads can be modified as follows:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-config-threads]
```

Basic authentication

Configuring basic authentication can be done by providing an `HttpClientConfigCallback` while building the `RestClient` through its builder. The interface has one method that receives an instance of `org.apache.http.impl.nio.client.HttpAsyncClientBuilder` as an argument and has the same

return type. The http client builder can be modified and then returned. In the following example we set a default credentials provider that requires basic authentication.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-config-basic-auth]
```

Preemptive Authentication can be disabled, which means that every request will be sent without authorization headers to see if it is accepted and, upon receiving a HTTP 401 response, it will resend the exact same request with the basic authentication header. If you wish to do this, then you can do so by disabling it via the `HttpAsyncClientBuilder`:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-config-disable-preemptive-auth]
```

① Disable preemptive authentication

Encrypted communication

Encrypted communication can also be configured through the `HttpClientConfigCallback`. The `org.apache.http.impl.nio.client.HttpAsyncClientBuilder` received as an argument exposes multiple methods to configure encrypted communication: `setSSLContext`, `setSSLSessionStrategy` and `setConnectionManager`, in order of precedence from the least important. The following is an example:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest
/src/test/java/org/elasticsearch/client/documentation/RestClientDocumentation.java[res
t-client-config-encrypted-communication]
```

If no explicit configuration is provided, the system default configuration will be used.

Others

For any other required configuration needed, the Apache HttpAsyncClient docs should be consulted: https://hc.apache.org/httpcomponents-asyncclient-4.1.x/ .

Sniffer

Minimal library that allows to automatically discover nodes from a running Elasticsearch cluster and set them to an existing `RestClient` instance. It retrieves by default the nodes that belong to the cluster using the Nodes Info api and uses jackson to parse the obtained json response.

Compatible with Elasticsearch 2.x and onwards.

Javadoc

The javadoc for the REST client sniffer can be found at https://artifacts.elastic.co/javadoc/org/elasticsearch/client/elasticsearch-rest-client-sniffer/6.1.1/index.html.

Maven Repository

The REST client sniffer is subject to the same release cycle as Elasticsearch. Replace the version with the desired sniffer version, first released with `5.0.0-alpha4`. There is no relation between the sniffer version and the Elasticsearch version that the client can communicate with. Sniffer supports fetching the nodes list from Elasticsearch 2.x and onwards.

Maven configuration

Here is how you can configure the dependency using maven as a dependency manager. Add the following to your `pom.xml` file:

```
<dependency>
    <groupId>org.elasticsearch.client</groupId>
    <artifactId>elasticsearch-rest-client-sniffer</artifactId>
    <version>6.1.1</version>
</dependency>
```

Gradle configuration

Here is how you can configure the dependency using gradle as a dependency manager. Add the following to your `build.gradle` file:

```
dependencies {
    compile 'org.elasticsearch.client:elasticsearch-rest-client-sniffer:6.1.1'
}
```

Usage

Once a `RestClient` instance has been created as shown in Initialization, a `Sniffer` can be associated to it. The `Sniffer` will make use of the provided `RestClient` to periodically (every 5 minutes by default) fetch the list of current nodes from the cluster and update them by calling `RestClient#setHosts`.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client
/sniffer/src/test/java/org/elasticsearch/client/sniff/documentation/SnifferDocumentati
on.java[sniffer-init]
```

It is important to close the `Sniffer` so that its background thread gets properly shutdown and all of

its resources are released. The `Sniffer` object should have the same lifecycle as the `RestClient` and get closed right before the client:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client
/sniffer/src/test/java/org/elasticsearch/client/sniff/documentation/SnifferDocumentati
on.java[sniffer-close]
```

The `Sniffer` updates the nodes by default every 5 minutes. This interval can be customized by providing it (in milliseconds) as follows:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client
/sniffer/src/test/java/org/elasticsearch/client/sniff/documentation/SnifferDocumentati
on.java[sniffer-interval]
```

It is also possible to enable sniffing on failure, meaning that after each failure the nodes list gets updated straightaway rather than at the following ordinary sniffing round. In this case a `SniffOnFailureListener` needs to be created at first and provided at `RestClient` creation. Also once the `Sniffer` is later created, it needs to be associated with that same `SniffOnFailureListener` instance, which will be notified at each failure and use the `Sniffer` to perform the additional sniffing round as described.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client
/sniffer/src/test/java/org/elasticsearch/client/sniff/documentation/SnifferDocumentati
on.java[sniff-on-failure]
```

① Set the failure listener to the `RestClient` instance

② When sniffing on failure, not only do the nodes get updated after each failure, but an additional sniffing round is also scheduled sooner than usual, by default one minute after the failure, assuming that things will go back to normal and we want to detect that as soon as possible. Said interval can be customized at `Sniffer` creation time through the `setSniffAfterFailureDelayMillis` method. Note that this last configuration parameter has no effect in case sniffing on failure is not enabled like explained above.

③ Set the `Sniffer` instance to the failure listener

The Elasticsearch Nodes Info api doesn't return the protocol to use when connecting to the nodes but only their `host:port` key-pair, hence `http` is used by default. In case `https` should be used instead, the `ElasticsearchHostsSniffer` instance has to be manually created and provided as follows:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client
/sniffer/src/test/java/org/elasticsearch/client/sniff/documentation/SnifferDocumentati
on.java[sniffer-https]
```

In the same way it is also possible to customize the `sniffRequestTimeout`, which defaults to one second. That is the `timeout` parameter provided as a querystring parameter when calling the Nodes

Info api, so that when the timeout expires on the server side, a valid response is still returned although it may contain only a subset of the nodes that are part of the cluster, the ones that have responded until then.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client
/sniffer/src/test/java/org/elasticsearch/client/sniff/documentation/SnifferDocumentati
on.java[sniff-request-timeout]
```

Also, a custom `HostsSniffer` implementation can be provided for advanced use-cases that may require fetching the hosts from external sources rather than from Elasticsearch:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client
/sniffer/src/test/java/org/elasticsearch/client/sniff/documentation/SnifferDocumentati
on.java[custom-hosts-sniffer]
```

① Fetch the hosts from the external source

License

Java High Level REST Client

added[6.0.0-beta1]

The Java High Level REST Client works on top of the Java Low Level REST client. Its main goal is to expose API specific methods, that accept request objects as an argument and return response objects, so that request marshalling and response un-marshalling is handled by the client itself.

Each API can be called synchronously or asynchronously. The synchronous methods return a response object, while the asynchronous methods, whose names end with the `async` suffix, require a listener argument that is notified (on the thread pool managed by the low level client) once a response or an error is received.

The Java High Level REST Client depends on the Elasticsearch core project. It accepts the same

request arguments as the `TransportClient` and returns the same response objects.

Getting started

This section describes how to get started with the high-level REST client from getting the artifact to using it in an application.

Compatibility

The Java High Level REST Client requires Java 1.8 and depends on the Elasticsearch core project. The client version is the same as the Elasticsearch version that the client was developed for. It accepts the same request arguments as the `TransportClient` and returns the same response objects. See the Migration Guide if you need to migrate an application from `TransportClient` to the new REST client.

The High Level Client is guaranteed to be able to communicate with any Elasticsearch node running on the same major version and greater or equal minor version. It doesn't need to be in the same minor version as the Elasticsearch nodes it communicates with, as it is forward compatible meaning that it supports communicating with later versions of Elasticsearch than the one it was developed for.

The 6.0 client is able to communicate with any 6.x Elasticsearch node, while the 6.1 client is for sure able to communicate with 6.1, 6.2 and any later 6.x version, but there may be incompatibility issues when communicating with a previous Elasticsearch node version, for instance between 6.1 and 6.0, in case the 6.1 client supports new request body fields for some APIs that are not known by the 6.0 node(s).

It is recommended to upgrade the High Level Client when upgrading the Elasticsearch cluster to a new major version, as REST API breaking changes may cause unexpected results depending on the node that is hit by the request, and newly added APIs will only be supported by the newer version of the client. The client should always be updated last, once all of the nodes in the cluster have been upgraded to the new major version.

Javadoc

The javadoc for the REST high level client can be found at https://artifacts.elastic.co/javadoc/org/elasticsearch/client/elasticsearch-rest-high-level-client/6.1.1/index.html.

Maven Repository

The high-level Java REST client is hosted on Maven Central. The minimum Java version required is `1.8`.

The High Level REST Client is subject to the same release cycle as Elasticsearch. Replace the version with the desired client version.

Maven configuration

Here is how you can configure the dependency using maven as a dependency manager. Add the following to your `pom.xml` file:

```
<dependency>
    <groupId>org.elasticsearch.client</groupId>
    <artifactId>elasticsearch-rest-high-level-client</artifactId>
    <version>6.1.1</version>
</dependency>
```

Gradle configuration

Here is how you can configure the dependency using gradle as a dependency manager. Add the following to your `build.gradle` file:

```
dependencies {
    compile 'org.elasticsearch.client:elasticsearch-rest-high-level-client:6.1.1'
}
```

Lucene Snapshot repository

The very first releases of any major version (like a beta), might have been built on top of a Lucene Snapshot version. In such a case you will be unable to resolve the Lucene dependencies of the client.

For example, if you want to use the `6.0.0-beta1` version which depends on Lucene `7.0.0-snapshot-00142c9`, you must define the following repository.

For Maven:

```
<repository>
    <id>elastic-lucene-snapshots</id>
    <name>Elastic Lucene Snapshots</name>
    <url>
http://s3.amazonaws.com/download.elasticsearch.org/lucenesnapshots/00142c9</url>
    <releases><enabled>true</enabled></releases>
    <snapshots><enabled>false</enabled></snapshots>
</repository>
```

For Gradle:

```
maven {
    url 'http://s3.amazonaws.com/download.elasticsearch.org/lucenesnapshots/00142c9'
}
```

Dependencies

The High Level Java REST Client depends on the following artifacts and their transitive dependencies:

- org.elasticsearch.client:elasticsearch-rest-client
- org.elasticsearch:elasticsearch

Initialization

A `RestHighLevelClient` instance needs a REST low-level client builder to be built as follows:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/MainDocumentationIT.java[rest-high-level-client-init]
```

The high-level client will internally create the low-level client used to perform requests based on the provided builder, and manage its lifecycle.

The high-level client instance needs to be closed when no longer needed so that all the resources used by it get properly released, as well as the underlying http client instance and its threads. This can be done through the `close` method, which will close the internal `RestClient` instance.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/MainDocumentationIT.java[rest-high-level-client-close]
```

In the rest of this documentation about the Java High Level Client, the `RestHighLevelClient` instance will be referenced as `client`.

Supported APIs

The Java High Level REST Client supports the following APIs:

Indices APIs

- Delete Index API

Single document APIs

- [java-rest-high-document-index]
- [java-rest-high-document-get]
- [java-rest-high-document-delete]
- Update API

Multi document APIs

- [java-rest-high-document-bulk]

Search APIs

- Search API
- Search Scroll API
- Clear Scroll API

Miscellaneous APIs

- [java-rest-high-main]

Delete Index API

Delete Index Request

A `DeleteIndexRequest` requires an `index` argument:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/IndicesClientDocumentationIT.java[delete-index-request]
```

① Index

Optional arguments

The following arguments can optionally be provided:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/IndicesClientDocumentationIT.java[delete-index-request-timeout]
```

① Timeout to wait for the all the nodes to acknowledge the index deletion as a `TimeValue`

② Timeout to wait for the all the nodes to acknowledge the index deletion as a `String`

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/IndicesClientDocumentationIT.java[delete-index-request-masterTimeout]
```

① Timeout to connect to the master node as a `TimeValue`

② Timeout to connect to the master node as a `String`

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/IndicesClientDocumentationIT.java[delete-index-request-indicesOptions]
```

① Setting `IndicesOptions` controls how unavailable indices are resolved and how wildcard expressions are expanded

Synchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation
/IndicesClientDocumentationIT.java[delete-index-execute]
```

Asynchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation
/IndicesClientDocumentationIT.java[delete-index-execute-async]
```

① Called when the execution is successfully completed. The response is provided as an argument

② Called in case of failure. The raised exception is provided as an argument

Delete Index Response

The returned `DeleteIndexResponse` allows to retrieve information about the executed operation as follows:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation
/IndicesClientDocumentationIT.java[delete-index-response]
```

① Indicates whether all of the nodes have acknowledged the request or not

If the index was not found, an `ElasticsearchException` will be thrown:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation
/IndicesClientDocumentationIT.java[delete-index-notfound]
```

① Do something if the index to be deleted was not found === Index API

Index Request

An `IndexRequest` requires the following arguments:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[index-request-string]
```

① Index

② Type

③ Document id

④ Document source provided as a `String`

Providing the document source

The document source can be provided in different ways:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-request-map]
```

① Document source provided as a `Map` which gets automatically converted to JSON format

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-request-xcontent]
```

① Document source provided as an `XContentBuilder` object, the Elasticsearch built-in helpers to generate JSON content

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-request-shortcut]
```

① Document source provided as `Object` key-pairs, which gets converted to JSON format

Optional arguments

The following arguments can optionally be provided:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-request-routing]
```

① Routing value

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-request-parent]
```

① Parent value

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-request-timeout]
```

① Timeout to wait for primary shard to become available as a `TimeValue`

② Timeout to wait for primary shard to become available as a `String`

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-request-refresh]
```

① Refresh policy as a `WriteRequest.RefreshPolicy` instance

② Refresh policy as a `String`

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-request-version]
```

① Version

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-request-version-type]
```

① Version type

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-request-op-type]
```

① Operation type provided as an `DocWriteRequest.OpType` value

② Operation type provided as a `String`: can be `create` or `update` (default)

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-request-pipeline]
```

① The name of the ingest pipeline to be executed before indexing the document

Synchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-execute]
```

Asynchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-execute-async]
```

① Called when the execution is successfully completed. The response is provided as an argument

② Called in case of failure. The raised exception is provided as an argument

Index Response

The returned `IndexResponse` allows to retrieve information about the executed operation as follows:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-response]
```

① Handle (if needed) the case where the document was created for the first time

② Handle (if needed) the case where the document was rewritten as it was already existing

③ Handle the situation where number of successful shards is less than total shards

④ Handle the potential failures

If there is a version conflict, an `ElasticsearchException` will be thrown:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-conflict]
```

① The raised exception indicates that a version conflict error was returned

Same will happen in case `opType` was set to `create` and a document with same index, type and id already existed:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[index-optype]
```

① The raised exception indicates that a version conflict error was returned === Get API

Get Request

A `GetRequest` requires the following arguments:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[get-request]
```

① Index

② Type

③ Document id

Optional arguments

The following arguments can optionally be provided:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[get-request-no-source]
```

① Disable source retrieval, enabled by default

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[get-request-source-include]
```

① Configure source inclusion for specific fields

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[get-request-source-exclude]
```

① Configure source exclusion for specific fields

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[get-request-stored]
```

① Configure retrieval for specific stored fields (requires fields to be stored separately in the mappings)

② Retrieve the `message` stored field (requires the field to be stored separately in the mappings)

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[get-request-routing]
```

① Routing value

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[get-request-parent]
```

① Parent value

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[get-request-preference]
```

① Preference value

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[get-request-realtime]
```

① Set realtime flag to `false` (`true` by default)

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[get-request-refresh]
```

① Perform a refresh before retrieving the document (`false` by default)

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[get-request-version]
```

① Version

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[get-request-version-type]
```

① Version type

Synchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[get-execute]
```

Asynchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[get-execute-async]
```

① Called when the execution is successfully completed. The response is provided as an argument.

② Called in case of failure. The raised exception is provided as an argument.

Get Response

The returned `GetResponse` allows to retrieve the requested document along with its metadata and eventually stored fields.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[get-response]
```

① Retrieve the document as a `String`

② Retrieve the document as a `Map<String, Object>`

③ Retrieve the document as a `byte[]`

④ Handle the scenario where the document was not found. Note that although the returned response has `404` status code, a valid `GetResponse` is returned rather than an exception thrown. Such response does not hold any source document and its `isExists` method returns `false`.

When a get request is performed against an index that does not exist, the response has `404` status code, an `ElasticsearchException` gets thrown which needs to be handled as follows:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[get-indexnotfound]
```

① Handle the exception thrown because the index does not exist

In case a specific document version has been requested, and the existing document has a different version number, a version conflict is raised:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[get-conflict]
```

① The raised exception indicates that a version conflict error was returned === Delete API

Delete Request

A `DeleteRequest` requires the following arguments:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[delete-request]
```

① Index

② Type

③ Document id

Optional arguments

The following arguments can optionally be provided:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[delete-request-routing]
```

① Routing value

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[delete-request-parent]
```

① Parent value

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[delete-request-timeout]
```

① Timeout to wait for primary shard to become available as a `TimeValue`

② Timeout to wait for primary shard to become available as a `String`

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[delete-request-refresh]
```

① Refresh policy as a `WriteRequest.RefreshPolicy` instance

② Refresh policy as a `String`

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[delete-request-version]
```

① Version

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[delete-request-version-type]
```

① Version type

Synchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[delete-execute]
```

Asynchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[delete-execute-async]
```

① Called when the execution is successfully completed. The response is provided as an argument

② Called in case of failure. The raised exception is provided as an argument

Delete Response

The returned `DeleteResponse` allows to retrieve information about the executed operation as follows:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[delete-response]
```

① Handle the situation where number of successful shards is less than total shards

② Handle the potential failures

It is also possible to check whether the document was found or not:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[delete-notfound]
```

① Do something if the document to be deleted was not found

If there is a version conflict, an `ElasticsearchException` will be thrown:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[delete-conflict]
```

① The raised exception indicates that a version conflict error was returned

Update API

Update Request

An `UpdateRequest` requires the following arguments:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[update-request]
```

① Index

② Type

③ Document id

The Update API allows to update an existing document by using a script or by passing a partial document.

Updates with a script

The script can be provided as an inline script:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[update-request-with-inline-script]
```

① Script parameters provided as a `Map` of objects

② Create an inline script using the `painless` language and the previous parameters

③ Sets the script to the update request

Or as a stored script:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-with-stored-script]
```

① Reference to a script stored under the name `increment-field` in the `painless` language

② Sets the script in the update request

Updates with a partial document

When using updates with a partial document, the partial document will be merged with the existing document.

The partial document can be provided in different ways:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-with-doc-as-string]
```

① Partial document source provided as a `String` in JSON format

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-with-doc-as-map]
```

① Partial document source provided as a `Map` which gets automatically converted to JSON format

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-with-doc-as-xcontent]
```

① Partial document source provided as an `XContentBuilder` object, the Elasticsearch built-in helpers to generate JSON content

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-shortcut]
```

① Partial document source provided as `Object` key-pairs, which gets converted to JSON format

Upserts

If the document does not already exist, it is possible to define some content that will be inserted as a new document using the `upsert` method:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-upsert]
```

① Upsert document source provided as a `String`

Similarly to the partial document updates, the content of the `upsert` document can be defined using methods that accept `String`, `Map`, `XContentBuilder` or `Object` key-pairs.

Optional arguments

The following arguments can optionally be provided:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-routing]
```

① Routing value

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-parent]
```

① Parent value

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-timeout]
```

① Timeout to wait for primary shard to become available as a `TimeValue`

② Timeout to wait for primary shard to become available as a `String`

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-refresh]
```

① Refresh policy as a `WriteRequest.RefreshPolicy` instance

② Refresh policy as a `String`

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-retry]
```

① How many times to retry the update operation if the document to update has been changed by another operation between the get and indexing phases of the update operation

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-no-source]
```

① Enable source retrieval, disabled by default

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-source-include]
```

① Configure source inclusion for specific fields

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-source-exclude]
```

① Configure source exclusion for specific fields

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-version]
```

① Version

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-detect-noop]
```

① Disable the noop detection

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-scripted-upsert]
```

① Indicate that the script must run regardless of whether the document exists or not, ie the script takes care of creating the document if it does not already exist.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT
.java[update-request-doc-upsert]
```

① Indicate that the partial document must be used as the upsert document if it does not exist yet.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[update-request-active-shards]
```

① Sets the number of shard copies that must be active before proceeding with the update operation.

② Number of shard copies provided as a `ActiveShardCount`: can be `ActiveShardCount.ALL`, `ActiveShardCount.ONE` or `ActiveShardCount.DEFAULT` (default)

Synchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[update-execute]
```

Asynchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[update-execute-async]
```

① Called when the execution is successfully completed. The response is provided as an argument.

② Called in case of failure. The raised exception is provided as an argument.

Update Response

The returned `UpdateResponse` allows to retrieve information about the executed operation as follows:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[update-response]
```

① Handle the case where the document was created for the first time (upsert)

② Handle the case where the document was updated

③ Handle the case where the document was deleted

④ Handle the case where the document was not impacted by the update, ie no operation (noop) was executed on the document

When the source retrieval is enabled in the `UpdateRequest` through the fetchSource method, the response contains the source of the updated document:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[update-getresult]
```

① Retrieve the updated document as a `GetResult`

② Retrieve the source of the updated document as a `String`

③ Retrieve the source of the updated document as a `Map<String, Object>`

④ Retrieve the source of the updated document as a `byte[]`

⑤ Handle the scenario where the source of the document is not present in the response (this is the case by default)

It is also possible to check for shard failures:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[update-failure]
```

① Handle the situation where number of successful shards is less than total shards

② Handle the potential failures

When a `UpdateRequest` is performed against a document that does not exist, the response has `404` status code, an `ElasticsearchException` gets thrown which needs to be handled as follows:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[update-docnotfound]
```

① Handle the exception thrown because the document not exist

If there is a version conflict, an `ElasticsearchException` will be thrown:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[update-conflict]
```

① The raised exception indicates that a version conflict error was returned. === Bulk API

NOTE	The Java High Level REST Client provides the Bulk Processor to assist with bulk requests

Bulk Request

A `BulkRequest` can be used to execute multiple index, update and/or delete operations using a single request.

It requires at least one operation to be added to the Bulk request:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-request]
```

① Creates the `BulkRequest`

② Adds a first `IndexRequest` to the Bulk request. See [java-rest-high-document-index] for more information on how to build `IndexRequest`.

③ Adds a second `IndexRequest`

④ Adds a third `IndexRequest`

WARNING	The Bulk API supports only documents encoded in JSON or SMILE. Providing documents in any other format will result in an error.

And different operation types can be added to the same `BulkRequest`:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-request-with-mixed-operations]
```

① Adds a `DeleteRequest` to the `BulkRequest`. See [java-rest-high-document-delete] for more information on how to build `DeleteRequest`.

② Adds an `UpdateRequest` to the `BulkRequest`. See Update API for more information on how to build `UpdateRequest`.

③ Adds an `IndexRequest` using the SMILE format

Optional arguments

The following arguments can optionally be provided:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-request-timeout]
```

① Timeout to wait for the bulk request to be performed as a `TimeValue`

② Timeout to wait for the bulk request to be performed as a `String`

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-request-refresh]
```

① Refresh policy as a `WriteRequest.RefreshPolicy` instance

② Refresh policy as a `String`

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-request-active-shards]
```

① Sets the number of shard copies that must be active before proceeding with the index/update/delete operations.

② Number of shard copies provided as a `ActiveShardCount`: can be `ActiveShardCount.ALL`, `ActiveShardCount.ONE` or `ActiveShardCount.DEFAULT` (default)

Synchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-execute]
```

Asynchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-execute-async]
```

① Called when the execution is successfully completed. The response is provided as an argument and contains a list of individual results for each operation that was executed. Note that one or more operations might have failed while the others have been successfully executed.

② Called when the whole `BulkRequest` fails. In this case the raised exception is provided as an argument and no operation has been executed.

Bulk Response

The returned `BulkResponse` contains information about the executed operations and allows to iterate over each result as follows:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-response]
```

① Iterate over the results of all operations

② Retrieve the response of the operation (successful or not), can be `IndexResponse`, `UpdateResponse` or `DeleteResponse` which can all be seen as `DocWriteResponse` instances

③ Handle the response of an index operation

④ Handle the response of a update operation

⑤ Handle the response of a delete operation

The Bulk response provides a method to quickly check if one or more operation has failed:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-has-failures]
```

① This method returns `true` if at least one operation failed

In such situation it is necessary to iterate over all operation results in order to check if the operation failed, and if so, retrieve the corresponding failure:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-errors]
```

① Indicate if a given operation failed

② Retrieve the failure of the failed operation

Bulk Processor

The `BulkProcessor` simplifies the usage of the Bulk API by providing a utility class that allows index/update/delete operations to be transparently executed as they are added to the processor.

In order to execute the requests, the `BulkProcessor` requires the following components:

`RestHighLevelClient`
: This client is used to execute the `BulkRequest` and to retrieve the `BulkResponse`

`BulkProcessor.Listener`
: This listener is called before and after every `BulkRequest` execution or when a `BulkRequest` failed

Then the `BulkProcessor.builder` method can be used to build a new `BulkProcessor`:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-processor-init]
```

① Create the `BulkProcessor.Listener`

② This method is called before each execution of a `BulkRequest`

③ This method is called after each execution of a `BulkRequest`

④ This method is called when a `BulkRequest` failed

⑤ Create the `BulkProcessor` by calling the `build()` method from the `BulkProcessor.Builder`. The `RestHighLevelClient.bulkAsync()` method will be used to execute the `BulkRequest` under the hood.

The `BulkProcessor.Builder` provides methods to configure how the `BulkProcessor` should handle requests execution:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-processor-options]
```

① Set when to flush a new bulk request based on the number of actions currently added (defaults to 1000, use -1 to disable it)

② Set when to flush a new bulk request based on the size of actions currently added (defaults to 5Mb, use -1 to disable it)

③ Set the number of concurrent requests allowed to be executed (default to 1, use 0 to only allow the execution of a single request)

④ Set a flush interval flushing any `BulkRequest` pending if the interval passes (defaults to not set)

⑤ Set a constant back off policy that initially waits for 1 second and retries up to 3 times. See `BackoffPolicy.noBackoff()`, `BackoffPolicy.constantBackoff()` and `BackoffPolicy.exponentialBackoff()` for more options.

Once the `BulkProcessor` is created requests can be added to it:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-processor-add]
```

The requests will be executed by the `BulkProcessor`, which takes care of calling the `BulkProcessor.Listener` for every bulk request.

The listener provides methods to access to the `BulkRequest` and the `BulkResponse`:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-processor-listener]
```

① Called before each execution of a `BulkRequest`, this method allows to know the number of operations that are going to be executed within the `BulkRequest`

② Called after each execution of a `BulkRequest`, this method allows to know if the `BulkResponse` contains errors

③ Called if the `BulkRequest` failed, this method allows to know the failure

Once all requests have been added to the `BulkProcessor`, its instance needs to be closed using one of the two available closing methods.

The `awaitClose()` method can be used to wait until all requests have been processed or the specified waiting time elapses:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-processor-await]
```

① The method returns `true` if all bulk requests completed and `false` if the waiting time elapsed before all the bulk requests completed

The `close()` method can be used to immediately close the `BulkProcessor`:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/CRUDDocumentationIT.java[bulk-processor-close]
```

Both methods flush the requests added to the processor before closing the processor and also forbid any new request to be added to it.

Search API

Search Request

The `SearchRequest` is used for any operation that has to do with searching documents, aggregations, suggestions and also offers ways of requesting highlighting on the resulting documents.

In its most basic form, we can add a query to the request:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-basic]
```

① Creates the `SeachRequest`. Without arguments this runs against all indices.

② Most search parameters are added to the `SearchSourceBuilder`. It offers setters for everything that goes into the search request body.

③ Add a `match_all` query to the `SearchSourceBuilder`.

Optional arguments

Let's first look at some of the optional arguments of a `SearchRequest`:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-indices-types]
```

① Restricts the request to an index

② Limits the request to a type

There are a couple of other interesting optional parameters:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-routing]
```

① Set a routing parameter

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-indicesOptions]
```

① Setting `IndicesOptions` controls how unavailable indices are resolved and how wildcard expressions are expanded

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-preference]
```

① Use the preference parameter e.g. to execute the search to prefer local shards. The default is to randomize across shards.

Using the SearchSourceBuilder

Most options controlling the search behavior can be set on the `SearchSourceBuilder`, which contains more or less the equivalent of the options in the search request body of the Rest API.

Here are a few examples of some common options:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-source-basics]
```

① Create a `SearchSourceBuilder` with default options.

② Set the query. Can be any type of `QueryBuilder`

③ Set the `from` option that determines the result index to start searching from. Defaults to 0.

④ Set the `size` option that determines the number of search hits to return. Defaults to 10.

⑤ Set an optional timeout that controls how long the search is allowed to take.

After this, the `SearchSourceBuilder` only needs to be added to the `SearchRequest`:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-source-setter]
```

Building queries

Search queries are created using `QueryBuilder` objects. A `QueryBuilder` exists for every search query type supported by Elasticsearch's {ref}/query-dsl.html[Query DSL].

A `QueryBuilder` can be created using its constructor:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-query-builder-ctor]
```

① Create a full text {ref}/query-dsl-match-query.html[Match Query] that matches the text "kimchy" over the field "user".

Once created, the `QueryBuilder` object provides methods to configure the options of the search query it creates:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-query-builder-options]
```

① Enable fuzzy matching on the match query

② Set the prefix length option on the match query

③ Set the max expansion options to control the fuzzy process of the query

`QueryBuilder` objects can also be created using the `QueryBuilders` utility class. This class provides helper methods that can be used to create `QueryBuilder` objects using a fluent programming style:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-query-builders]
```

Whatever the method used to create it, the `QueryBuilder` object must be added to the `SearchSourceBuilder` as follows:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-query-setter]
```

The Building Queries page gives a list of all available search queries with their corresponding `QueryBuilder` objects and `QueryBuilders` helper methods.

Specifying Sorting

The `SearchSourceBuilder` allows to add one or more `SortBuilder` instances. There are four special implementations (Field-, Score-, GeoDistance- and ScriptSortBuilder).

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-source-sorting]
```

① Sort descending by `_score` (the default)

② Also sort ascending by `_id` field

Source filtering

By default, search requests return the contents of the document `_source` but like in the Rest API you can overwrite this behavior. For example, you can turn off `_source` retrieval completely:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-source-filtering-off]
```

The method also accepts an array of one or more wildcard patterns to control which fields get included or excluded in a more fine grained way:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-source-filtering-includes]
```

Requesting Highlighting

Highlighting search results can be achieved by setting a `HighlightBuilder` on the `SearchSourceBuilder`. Different highlighting behaviour can be defined for each fields by adding one or more `HighlightBuilder.Field` instances to a `HighlightBuilder`.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-highlighting]
```

① Creates a new `HighlightBuilder`

② Create a field highlighter for the `title` field

③ Set the field highlighter type

④ Add the field highlighter to the highlight builder

There are many options which are explained in detail in the Rest API documentation. The Rest API parameters (e.g. `pre_tags`) are usually changed by setters with a similar name (e.g. `#preTags(String …)`).

Highlighted text fragments can later be retrieved from the `SearchResponse`.

Requesting Aggregations

Aggregations can be added to the search by first creating the appropriate `AggregationBuilder` and then setting it on the `SearchSourceBuilder`. In the following example we create a `terms` aggregation on company names with a sub-aggregation on the average age of employees in the company:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-aggregations]
```

The Building Aggregations page gives a list of all available aggregations with their corresponding `AggregationBuilder` objects and `AggregationBuilders` helper methods.

We will later see how to access aggregations in the `SearchResponse`.

Requesting Suggestions

To add Suggestions to the search request, use one of the `SuggestionBuilder` implementations that are easily accessible from the `SuggestBuilders` factory class. Suggestion builders need to be added to the top level `SuggestBuilder`, which itself can be set on the `SearchSourceBuilder`.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-suggestion]
```

① Creates a new `TermSuggestionBuilder` for the `user` field and the text `kmichy`

② Adds the suggestion builder and names it `suggest_user`

We will later see how to retrieve suggestions from the `SearchResponse`.

Profiling Queries and Aggregations

The {ref}/search-profile.html[Profile API] can be used to profile the execution of queries and aggregations for a specific search request. in order to use it, the profile flag must be set to true on the `SearchSourceBuilder`:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-profiling]
```

Once the `SearchRequest` is executed the corresponding `SearchResponse` will contain the profiling results.

Synchronous Execution

When executing a `SearchRequest` in the following manner, the client waits for the `SearchResponse` to be returned before continuing with code execution:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-execute]
```

Asynchronous Execution

Executing a `SearchRequest` can also be done in an asynchronous fashion so that the client can return directly. Users need to specify how the response or potential failures will be handled by passing in appropriate listeners:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-execute-async]
```

① Called when the execution is successfully completed.

② Called when the whole `SearchRequest` fails.

SearchResponse

The `SearchResponse` that is returned by executing the search provides details about the search execution itself as well as access to the documents returned. First, there is useful information about the request execution itself, like the HTTP status code, execution time or wether the request terminated early or timed out:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-response-1]
```

Second, the response also provides information about the execution on the shard level by offering statistics about the total number of shards that were affected by the search, and the successful vs. unsuccessful shards. Possible failures can also be handled by iterating over an array off `ShardSearchFailures` like in the following example:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-response-2]
```

Retrieving SearchHits

To get access to the returned documents, we need to first get the `SearchHits` contained in the response:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT
.java[search-hits-get]
```

The `SearchHits` provides global information about all hits, like total number of hits or the maximum score:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT
.java[search-hits-info]
```

Nested inside the `SearchHits` are the individual search results that can be iterated over:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT
.java[search-hits-singleHit]
```

The `SearchHit` provides access to basic information like index, type, docId and score of each search hit:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT
.java[search-hits-singleHit-properties]
```

Furthermore, it lets you get back the document source, either as a simple JSON-String or as a map of key/value pairs. In this map, regular fields are keyed by the field name and contain the field value. Multi-valued fields are returned as lists of objects, nested objects as another key/value map. These cases need to be cast accordingly:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT
.java[search-hits-singleHit-source]
```

Retrieving Highlighting

If requested, highlighted text fragments can be retrieved from each `SearchHit` in the result. The hit object offers access to a map of field names to `HighlightField` instances, each of which contains one or many highlighted text fragments:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT
.java[search-request-highlighting-get]
```

① Get the highlighting for the `title` field

② Get one or many fragments containing the highlighted field content

Retrieving Aggregations

Aggregations can be retrieved from the `SearchResponse` by first getting the root of the aggregation tree, the `Aggregations` object, and then getting the aggregation by name.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-aggregations-get]
```

① Get the `by_company` terms aggregation

② Get the buckets that is keyed with `Elastic`

③ Get the `average_age` sub-aggregation from that bucket

Note that if you access aggregations by name, you need to specify the aggregation interface according to the type of aggregation you requested, otherwise a `ClassCastException` will be thrown:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-aggregations-get-wrongCast]
```

① This will throw an exception because "by_company" is a `terms` aggregation but we try to retrieve it as a `range` aggregation

It is also possible to access all aggregations as a map that is keyed by the aggregation name. In this case, the cast to the proper aggregation interface needs to happen explicitly:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-aggregations-asMap]
```

There are also getters that return all top level aggregations as a list:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-aggregations-asList]
```

And last but not least you can iterate over all aggregations and then e.g. decide how to further process them based on their type:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-aggregations-iterator]
```

Retrieving Suggestions

To get back the suggestions from a `SearchResponse`, use the `Suggest` object as an entry point and then retrieve the nested suggestion objects:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-suggestion-get]
```

① Use the `Suggest` class to access suggestions

② Suggestions can be retrieved by name. You need to assign them to the correct type of Suggestion class (here `TermSuggestion`), otherwise a `ClassCastException` is thrown

③ Iterate over the suggestion entries

④ Iterate over the options in one entry

Retrieving Profiling Results

Profiling results are retrieved from a `SearchResponse` using the `getProfileResults()` method. This method returns a `Map` containing a `ProfileShardResult` object for every shard involved in the `SearchRequest` execution. `ProfileShardResult` are stored in the `Map` using a key that uniquely identifies the shard the profile result corresponds to.

Here is a sample code that shows how to iterate over all the profiling results of every shard:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-profiling-get]
```

① Retrieve the `Map` of `ProfileShardResult` from the `SearchResponse`

② Profiling results can be retrieved by shard's key if the key is known, otherwise it might be simpler to iterate over all the profiling results

③ Retrieve the key that identifies which shard the `ProfileShardResult` belongs to

④ Retrieve the `ProfileShardResult` for the given shard

The `ProfileShardResult` object itself contains one or more query profile results, one for each query executed against the underlying Lucene index:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-profiling-queries]
```

① Retrieve the list of `QueryProfileShardResult`

② Iterate over each `QueryProfileShardResult`

Each `QueryProfileShardResult` gives access to the detailed query tree execution, returned as a list of `ProfileResult` objects:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-profiling-queries-results]
```

① Iterate over the profile results

② Retrieve the name of the Lucene query

③ Retrieve the time in millis spent executing the Lucene query

④ Retrieve the profile results for the sub-queries (if any)

The Rest API documentation contains more information about {ref}/_profiling_queries.html[Profiling Queries] with a description of the {ref}/_profiling_queries.html#_literal_query_literal_section[query profiling information]

The `QueryProfileShardResult` also gives access to the profiling information for the Lucene collectors:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-profiling-queries-collectors]
```

① Retrieve the profiling result of the Lucene collector

② Retrieve the name of the Lucene collector

③ Retrieve the time in millis spent executing the Lucene collector

④ Retrieve the profile results for the sub-collectors (if any)

The Rest API documentation contains more information about profiling information {ref}/_profiling_queries.html#_literal_collectors_literal_section[for Lucene collectors].

In a very similar manner to the query tree execution, the `QueryProfileShardResult` objects gives access to the detailed aggregations tree execution:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-request-profiling-aggs]
```

① Retrieve the `AggregationProfileShardResult`

② Iterate over the aggregation profile results

③ Retrieve the type of the aggregation (corresponds to Java class used to execute the aggregation)

④ Retrieve the time in millis spent executing the Lucene collector

⑤ Retrieve the profile results for the sub-aggregations (if any)

The Rest API documentation contains more information about {ref}/_profiling_aggregations.html[Profiling Aggregations]

Search Scroll API

The Scroll API can be used to retrieve a large number of results from a search request.

In order to use scrolling, the following steps need to be executed in the given order.

Initialize the search scroll context

An initial search request with a `scroll` parameter must be executed to initialize the scroll session through the Search API. When processing this `SearchRequest`, Elasticsearch detects the presence of the `scroll` parameter and keeps the search context alive for the corresponding time interval.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-scroll-init]
```

① Create the `SearchRequest` and its corresponding `SearchSourceBuilder`. Also optionally set the `size` to control how many results to retrieve at a time.

② Set the scroll interval

③ Read the returned scroll id, which points to the search context that's being kept alive and will be needed in the following search scroll call

④ Retrieve the first batch of search hits

Retrieve all the relevant documents

As a second step, the received scroll identifier must be set to a `SearchScrollRequest` along with a new scroll interval and sent through the `searchScroll` method. Elasticsearch returns another batch of results with a new scroll identifier. This new scroll identifier can then be used in a subsequent `SearchScrollRequest` to retrieve the next batch of results, and so on. This process should be repeated in a loop until no more results are returned, meaning that the scroll has been exhausted and all the matching documents have been retrieved.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-scroll2]
```

① Create the `SearchScrollRequest` by setting the required scroll id and the scroll interval

② Read the new scroll id, which points to the search context that's being kept alive and will be needed in the following search scroll call

③ Retrieve another batch of search hits <4>

Clear the scroll context

Finally, the last scroll identifier can be deleted using the Clear Scroll API in order to release the search context. This happens automatically when the scroll expires, but it's good practice to do it as soon as the scroll session is completed.

Optional arguments

The following arguments can optionally be provided when constructing the `SearchScrollRequest`:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[scroll-request-arguments]
```

① Scroll interval as a `TimeValue`

② Scroll interval as a `String`

If no `scroll` value is set for the `SearchScrollRequest`, the search context will expire once the initial scroll time expired (ie, the scroll time set in the initial search request).

Synchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-scroll-execute-sync]
```

Asynchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-scroll-execute-async]
```

① Called when the execution is successfully completed. The response is provided as an argument

② Called in case of failure. The raised exception is provided as an argument

Response

The search scroll API returns a `SearchResponse` object, same as the Search API.

Full example

The following is a complete example of a scrolled search.

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[search-scroll-example]
```

① Initialize the search context by sending the initial `SearchRequest`

② Retrieve all the search hits by calling the Search Scroll api in a loop until no documents are returned

③ Create a new `SearchScrollRequest` holding the last returned scroll identifier and the scroll interval

④ Process the returned search results

⑤ Clear the scroll context once the scroll is completed

Clear Scroll API

The search contexts used by the Search Scroll API are automatically deleted when the scroll times out. But it is advised to release search contexts as soon as they are not necessary anymore using the Clear Scroll API.

Clear Scroll Request

A `ClearScrollRequest` can be created as follows:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[clear-scroll-request]
```

① Create a new `ClearScrollRequest`

② Adds a scroll id to the list of scroll identifiers to clear

Providing the scroll identifiers

The `ClearScrollRequest` allows to clear one or more scroll identifiers in a single request.

The scroll identifiers can be added to the request one by one:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[clear-scroll-add-scroll-id]
```

Or all together using:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[clear-scroll-add-scroll-ids]
```

Synchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[clear-scroll-execute]
```

Asynchronous Execution

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[clear-scroll-execute-async]
```

① Called when the execution is successfully completed. The response is provided as an argument

② Called in case of failure. The raised exception is provided as an argument

Clear Scroll Response

The returned `ClearScrollResponse` allows to retrieve information about the released search contexts:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/SearchDocumentationIT.java[clear-scroll-response]
```

① Return true if the request succeeded

② Return the number of released search contexts === Info API

Execution

Cluster information can be retrieved using the `info()` method:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/MainDocumentationIT.java[main-execute]
```

Response

The returned `MainResponse` provides various kinds of information about the cluster:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/MainDocumentationIT.java[main-response]
```

① Retrieve the name of the cluster as a `ClusterName`

② Retrieve the unique identifier of the cluster

③ Retrieve the name of the node the request has been executed on

④ Retrieve the version of the node the request has been executed on

⑤ Retrieve the build information of the node the request has been executed on == Using Java Builders

The Java High Level REST Client depends on the Elasticsearch core project which provides different types of Java `Builders` objects, including:

Query Builders

The query builders are used to create the query to execute within a search request. There is a query builder for every type of query supported by the Query DSL. Each query builder implements the `QueryBuilder` interface and allows to set the specific options for a given type of query. Once created, the `QueryBuilder` object can be set as the query parameter of `SearchSourceBuilder`. The Search Request page shows an example of how to build a full search request using `SearchSourceBuilder` and `QueryBuilder` objects. The Building Search Queries page gives a list of all available search queries with their corresponding `QueryBuilder` objects and `QueryBuilders` helper methods.

Aggregation Builders

Similarly to query builders, the aggregation builders are used to create the aggregations to compute during a search request execution. There is an aggregation builder for every type of aggregation (or pipeline aggregation) supported by Elasticsearch. All builders extend the `AggregationBuilder` class (or `PipelineAggregationBuilder`class). Once created, `AggregationBuilder` objects can be set as the aggregation parameter of `SearchSourceBuilder`. There is a example of how `AggregationBuilder` objects are used with `SearchSourceBuilder` objects to define the aggregations to compute with a search query in Search Request page. The Building Aggregations page gives a list of all available aggregations with their corresponding `AggregationBuilder` objects and `AggregationBuilders` helper methods.

Building Queries

This page lists all the available search queries with their corresponding `QueryBuilder` class name and helper method name in the `QueryBuilders` utility class.

Match All Query

Search Query	QueryBuilder Class	Method in QueryBuilders
{ref}/query-dsl-match-all-query.html[Match All]	MatchAllQueryBuilder	QueryBuilders.matchAllQuery()

Full Text Queries

Search Query	QueryBuilder Class	Method in QueryBuilders
{ref}/query-dsl-match-query.html[Match]	MatchQueryBuilder	QueryBuilders.matchQuery()
{ref}/query-dsl-match-query-phrase.html[Match Phrase]	MatchPhraseQueryBuilder	QueryBuilders.matchPhraseQuery()
{ref}/query-dsl-match-query-phrase-prefix.html[Match Phrase Prefix]	MatchPhrasePrefixQueryBuilder	QueryBuilders.matchPhrasePrefixQuery()
{ref}/query-dsl-multi-match-query.html[Multi Match]	MultiMatchQueryBuilder	QueryBuilders.multiMatchQuery()
{ref}/query-dsl-common-terms-query.html[Common Terms]	CommonTermsQueryBuilder	QueryBuilders.commonTermsQuery()
{ref}/query-dsl-query-string-query.html[Query String]	QueryStringQueryBuilder	QueryBuilders.queryStringQuery()
{ref}/query-dsl-simple-query-string-query.html[Simple Query String]	SimpleQueryStringBuilder	QueryBuilders.simpleQueryStringQuery()

Term level queries

Search Query	QueryBuilder Class	Method in QueryBuilders
{ref}/query-dsl-term-query.html[Term]	TermQueryBuilder	QueryBuilders.termQuery()
{ref}/query-dsl-terms-query.html[Terms]	TermsQueryBuilder	QueryBuilders.termsQuery()
{ref}/query-dsl-range-query.html[Range]	RangeQueryBuilder	QueryBuilders.rangeQuery()
{ref}/query-dsl-exists-query.html[Exists]	ExistsQueryBuilder	QueryBuilders.existsQuery()
{ref}/query-dsl-prefix-query.html[Prefix]	PrefixQueryBuilder	QueryBuilders.prefixQuery()
{ref}/query-dsl-wildcard-query.html[Wildcard]	WildcardQueryBuilder	QueryBuilders.wildcardQuery()
{ref}/query-dsl-regexp-query.html[Regexp]	RegexpQueryBuilder	QueryBuilders.regexpQuery()
{ref}/query-dsl-fuzzy-query.html[Fuzzy]	FuzzyQueryBuilder	QueryBuilders.fuzzyQuery()
{ref}/query-dsl-type-query.html[Type]	TypeQueryBuilder	QueryBuilders.typeQuery()

Search Query	QueryBuilder Class	Method in QueryBuilders
{ref}/query-dsl-ids-query.html[Ids]	IdsQueryBuilder	QueryBuilders.idsQuery()

Compound queries

Search Query	QueryBuilder Class	Method in QueryBuilders
{ref}/query-dsl-constant-score-query.html[Constant Score]	ConstantScoreQueryBuilder	QueryBuilders.constantScoreQuery()
{ref}/query-dsl-bool-query.html[Bool]	BoolQueryBuilder	QueryBuilders.boolQuery()
{ref}/query-dsl-dis-max-query.html[Dis Max]	DisMaxQueryBuilder	QueryBuilders.disMaxQuery()
{ref}/query-dsl-function-score-query.html[Function Score]	FunctionScoreQueryBuilder	QueryBuilders.functionScoreQuery()
{ref}/query-dsl-boosting-query.html[Boosting]	BoostingQueryBuilder	QueryBuilders.boostingQuery()

Joining queries

Search Query	QueryBuilder Class	Method in QueryBuilders
{ref}/query-dsl-nested-query.html[Nested]	NestedQueryBuilder	QueryBuilders.nestedQuery()
{ref}/query-dsl-has-child-query.html[Has Child]	HasChildQueryBuilder	
{ref}/query-dsl-has-parent-query.html[Has Parent]	HasParentQueryBuilder	
{ref}/query-dsl-parent-id-query.html[Parent Id]	ParentIdQueryBuilder	

Geo queries

Search Query	QueryBuilder Class	Method in QueryBuilders
{ref}/query-dsl-geo-shape-query.html[GeoShape]	GeoShapeQueryBuilder	QueryBuilders.geoShapeQuery()
{ref}/query-dsl-geo-bounding-box-query.html[Geo Bounding Box]	GeoBoundingBoxQueryBuilder	QueryBuilders.geoBoundingBoxQuery()
{ref}/query-dsl-geo-distance-query.html[Geo Distance]	GeoDistanceQueryBuilder	QueryBuilders.geoDistanceQuery()

Search Query	QueryBuilder Class	Method in QueryBuilders
{ref}/query-dsl-geo-polygon-query.html[Geo Polygon]	GeoPolygonQueryBuilder	QueryBuilders.geoPolygonQuery()

Specialized queries

Search Query	QueryBuilder Class	Method in QueryBuilders
{ref}/query-dsl-mlt-query.html[More Like This]	MoreLikeThisQueryBuilder	QueryBuilders.moreLikeThisQuery()
{ref}/query-dsl-script-query.html[Script]	ScriptQueryBuilder	QueryBuilders.scriptQuery()
{ref}/query-dsl-percolate-query.html[Percolate]	PercolateQueryBuilder	

Span queries

Search Query	QueryBuilder Class	Method in QueryBuilders
{ref}/query-dsl-span-term-query.html[Span Term]	SpanTermQueryBuilder	QueryBuilders.spanTermQuery()
{ref}/query-dsl-span-multi-term-query.html[Span Multi Term]	SpanMultiTermQueryBuilder	QueryBuilders.spanMultiTermQueryBuilder()
{ref}/query-dsl-span-first-query.html[Span First]	SpanFirstQueryBuilder	QueryBuilders.spanFirstQuery()
{ref}/query-dsl-span-near-query.html[Span Near]	SpanNearQueryBuilder	QueryBuilders.spanNearQuery()
{ref}/query-dsl-span-or-query.html[Span Or]	SpanOrQueryBuilder	QueryBuilders.spanOrQuery()
{ref}/query-dsl-span-not-query.html[Span Not]	SpanNotQueryBuilder	QueryBuilders.spanNotQuery()
{ref}/query-dsl-span-containing-query.html[Span Containing]	SpanContainingQueryBuilder	QueryBuilders.spanContainingQuery()
{ref}/query-dsl-span-within-query.html[Span Within]	SpanWithinQueryBuilder	QueryBuilders.spanWithinQuery()
{ref}/query-dsl-span-field-masking-query.html[Span Field Masking]	FieldMaskingSpanQueryBuilder	QueryBuilders.fieldMaskingSpanQuery()

Building Aggregations

This page lists all the available aggregations with their corresponding `AggregationBuilder` class name and helper method name in the `AggregationBuilders` or `PipelineAggregatorBuilders` utility classes.

Metrics Aggregations

Aggregation	AggregationBuilder Class	Method in AggregationBuilders
{ref}/search-aggregations-metrics-avg-aggregation.html[Avg]	AvgAggregationBuilder	AggregationBuilders.avg()
{ref}/search-aggregations-metrics-cardinality-aggregation.html[Cardinality]	CardinalityAggregationBuilder	AggregationBuilders.cardinality()
{ref}/search-aggregations-metrics-extendedstats-aggregation.html[Extended Stats]	ExtendedStatsAggregationBuilder	AggregationBuilders.extendedStats()
{ref}/search-aggregations-metrics-geobounds-aggregation.html[Geo Bounds]	GeoBoundsAggregationBuilder	AggregationBuilders.geoBounds()
{ref}/search-aggregations-metrics-geocentroid-aggregation.html[Geo Centroid]	GeoCentroidAggregationBuilder	AggregationBuilders.geoCentroid()
{ref}/search-aggregations-metrics-max-aggregation.html[Max]	MaxAggregationBuilder	AggregationBuilders.max()
{ref}/search-aggregations-metrics-min-aggregation.html[Min]	MinxAggregationBuilder	AggregationBuilders.min()
{ref}/search-aggregations-metrics-percentile-aggregation.html[Percentiles]	PercentilesAggregationBuilder	AggregationBuilders.percentiles()
{ref}/search-aggregations-metrics-percentile-rank-aggregation.html[Percentile Ranks]	PercentileRanksAggregationBuilder	AggregationBuilders.percentileRanks()
{ref}/search-aggregations-metrics-scripted-metric-aggregation.html[Scripted Metric]	ScriptedMetricAggregationBuilder	AggregationBuilders.scriptedMetric()

Aggregation	AggregationBuilder Class	Method in AggregationBuilders
{ref}/search-aggregations-metrics-stats-aggregation.html[Stats]	StatsAggregationBuilder	AggregationBuilders.stats()
{ref}/search-aggregations-metrics-sum-aggregation.html[Sum]	SumAggregationBuilder	AggregationBuilders.sum()
{ref}/search-aggregations-metrics-top-hits-aggregation.html[Top hits]	TopHitsAggregationBuilder	AggregationBuilders.topHits()
{ref}/search-aggregations-metrics-valuecount-aggregation.html[Value Count]	ValueCountAggregationBuilder	AggregationBuilders.count()

Bucket Aggregations

Aggregation	AggregationBuilder Class	Method in AggregationBuilders
{ref}/search-aggregations-bucket-adjacency-matrix-aggregation.html[Adjacency Matrix]	AdjacencyMatrixAggregationBuilder	AggregationBuilders.adjacencyMatrix()
{ref}/search-aggregations-bucket-children-aggregation.html[Children]	ChildrenAggregationBuilder	
{ref}/search-aggregations-bucket-datehistogram-aggregation.html[Date Histogram]	DateHistogramAggregationBuilder	AggregationBuilders.dateHistogram()
{ref}/search-aggregations-bucket-daterange-aggregation.html[Date Range]	DateRangeAggregationBuilder	AggregationBuilders.dateRange()
{ref}/search-aggregations-bucket-diversified-sampler-aggregation.html[Diversified Sampler]	DiversifiedAggregationBuilder	AggregationBuilders.diversifiedSampler()
{ref}/search-aggregations-bucket-filter-aggregation.html[Filter]	FilterAggregationBuilder	AggregationBuilders.filter()
{ref}/search-aggregations-bucket-filters-aggregation.html[Filters]	FiltersAggregationBuilder	AggregationBuilders.filters()

Aggregation	AggregationBuilder Class	Method in AggregationBuilders
{ref}/search-aggregations-bucket-geodistance-aggregation.html[Geo Distance]	GeoDistanceAggregationBuilder	AggregationBuilders.geoDistance()
{ref}/search-aggregations-bucket-geohashgrid-aggregation.html[GeoHash Grid]	GeoGridAggregationBuilder	AggregationBuilders.geohashGrid()
{ref}/search-aggregations-bucket-global-aggregation.html[Global]	GlobalAggregationBuilder	AggregationBuilders.global()
{ref}/search-aggregations-bucket-histogram-aggregation.html[Histogram]	HistogramAggregationBuilder	AggregationBuilders.histogram()
{ref}/search-aggregations-bucket-iprange-aggregation.html[IP Range]	IpRangeAggregationBuilder	AggregationBuilders.ipRange()
{ref}/search-aggregations-bucket-missing-aggregation.html[Missing]	MissingAggregationBuilder	AggregationBuilders.missing()
{ref}/search-aggregations-bucket-nested-aggregation.html[Nested]	NestedAggregationBuilder	AggregationBuilders.nested()
{ref}/search-aggregations-bucket-range-aggregation.html[Range]	RangeAggregationBuilder	AggregationBuilders.range()
{ref}/search-aggregations-bucket-reverse-nested-aggregation.html[Reverse nested]	ReverseNestedAggregationBuilder	AggregationBuilders.reverseNested()
{ref}/search-aggregations-bucket-sampler-aggregation.html[Sampler]	SamplerAggregationBuilder	AggregationBuilders.sampler()
{ref}/search-aggregations-bucket-significantterms-aggregation.html[Significant Terms]	SignificantTermsAggregationBuilder	AggregationBuilders.significantTerms()
{ref}/search-aggregations-bucket-significanttext-aggregation.html[Significant Text]	SignificantTextAggregationBuilder	AggregationBuilders.significantText()

Aggregation	AggregationBuilder Class	Method in AggregationBuilders
{ref}/search-aggregations-bucket-terms-aggregation.html[Terms]	TermsAggregationBuilder	AggregationBuilders.terms()

Pipeline Aggregations

Pipeline on	PipelineAggregationBuilder Class	Method in PipelineAggregatorBuilders
{ref}/search-aggregations-pipeline-avg-bucket-aggregation.html[Avg Bucket]	AvgBucketPipelineAggregationBuilder	PipelineAggregatorBuilders.avgBucket()
{ref}/search-aggregations-pipeline-derivative-aggregation.html[Derivative]	DerivativePipelineAggregationBuilder	PipelineAggregatorBuilders.derivative()
{ref}/search-aggregations-pipeline-max-bucket-aggregation.html[Max Bucket]	MaxBucketPipelineAggregationBuilder	PipelineAggregatorBuilders.maxBucket()
{ref}/search-aggregations-pipeline-min-bucket-aggregation.html[Min Bucket]	MinBucketPipelineAggregationBuilder	PipelineAggregatorBuilders.minBucket()
{ref}/search-aggregations-pipeline-sum-bucket-aggregation.html[Sum Bucket]	SumBucketPipelineAggregationBuilder	PipelineAggregatorBuilders.sumBucket()
{ref}/search-aggregations-pipeline-stats-bucket-aggregation.html[Stats Bucket]	StatsBucketPipelineAggregationBuilder	PipelineAggregatorBuilders.statsBucket()
{ref}/search-aggregations-pipeline-extended-stats-bucket-aggregation.html[Extended Stats Bucket]	ExtendedStatsBucketPipelineAggregationBuilder	PipelineAggregatorBuilders.extendedStatsBucket()
{ref}/search-aggregations-pipeline-percentiles-bucket-aggregation.html[Percentiles Bucket]	PercentilesBucketPipelineAggregationBuilder	PipelineAggregatorBuilders.percentilesBucket()
{ref}/search-aggregations-pipeline-movavg-aggregation.html[Moving Average]	MovAvgPipelineAggregationBuilder	PipelineAggregatorBuilders.movingAvg()

Pipeline on	PipelineAggregationBuilder Class	Method in PipelineAggregatorBuilders
{ref}/search-aggregations-pipeline-cumulative-sum-aggregation.html[Cumulative Sum]	CumulativeSumPipelineAggregationBuilder	PipelineAggregatorBuilders.cumulativeSum()
{ref}/search-aggregations-pipeline-bucket-script-aggregation.html[Bucket Script]	BucketScriptPipelineAggregationBuilder	PipelineAggregatorBuilders.bucketScript()
{ref}/search-aggregations-pipeline-bucket-selector-aggregation.html[Bucket Selector]	BucketSelectorPipelineAggregationBuilder	PipelineAggregatorBuilders.bucketSelector()
{ref}/search-aggregations-pipeline-serialdiff-aggregation.html[Serial Differencing]	SerialDiffPipelineAggregationBuilder	PipelineAggregatorBuilders.diff()

Matrix Aggregations

Aggregation	AggregationBuilder Class	Method in MatrixStatsAggregationBuilders
{ref}/search-aggregations-matrix-stats-aggregation.html[Matrix Stats]	MatrixStatsAggregationBuilder	MatrixStatsAggregationBuilders.matrixStats()

Migration Guide

This section describes how to migrate existing code from the `TransportClient` to the new Java High Level REST Client released with the version 5.6.0 of Elasticsearch.

Motivations around a new Java client

The existing `TransportClient` has been part of Elasticsearch since its very first commit. It is a special client as it uses the transport protocol to communicate with Elasticsearch, which causes compatibility problems if the client is not on the same version as the Elasticsearch instances it talks to.

We released a low-level REST client in 2016, which is based on the well known Apache HTTP client and it allows to communicate with an Elasticsearch cluster in any version using HTTP. On top of that we released the high-level REST client which is based on the low-level client but takes care of request marshalling and response un-marshalling.

If you're interested in knowing more about these changes, we wrote a blog post about the state of

the official Elasticsearch Java clients.

Prerequisite

The Java High Level Rest Client requires Java `1.8` and can be used to send requests to an Elasticsearch cluster in a compatible version.

How to migrate

Adapting existing code to use the `RestHighLevelClient` instead of the `TransportClient` requires the following steps:

- Update dependencies
- Update client initialization
- Update application code

Updating the dependencies

Java application that uses the `TransportClient` depends on the `org.elasticsearch.client:transport` artifact. This dependency must be replaced by a new dependency on the high-level client.

The Getting Started page shows typical configurations for Maven and Gradle and presents the dependencies brought by the high-level client.

Changing the client's initialization code

The `TransportClient` is typically initialized as follows:

```
Settings settings = Settings.builder()
        .put("cluster.name", "prod").build();

TransportClient transportClient = new PreBuiltTransportClient(settings)
        .addTransportAddress(new TransportAddress(InetAddress.getByName("localhost"),
9300))
        .addTransportAddress(new TransportAddress(InetAddress.getByName("localhost"),
9301));
```

The initialization of a `RestHighLevelClient` is different. It requires to provide a low-level client builder as a constructor argument:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/MainDocumentationIT
.java[rest-high-level-client-init]
```

NOTE	The `RestClient` uses Elasticsearch's HTTP service which is bounded by default on `9200`. This port is different from the port used to connect to Elasticsearch with a `TransportClient`.

The `RestHighLevelClient` is thread-safe. It is typically instantiated by the application at startup time or when the first request is executed.

Once the `RestHighLevelClient` is initialized, it can be used to execute any of the supported APIs.

As with the `TransportClient`, the `RestHighLevelClient` must be closed when it is not needed anymore or when the application is stopped.

The code that closes the `TransportClient`:

```
transportClient.close();
```

must be replaced with:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation/MainDocumentationIT
.java[rest-high-level-client-close]
```

Changing the application's code

The `RestHighLevelClient` supports the same request and response objects as the `TransportClient`, but exposes slightly different methods to send the requests.

More importantly, the high-level client:

- does not support request builders. The legacy methods like `client.prepareIndex()` must be changed to use request constructors like `new IndexRequest()` to create requests objects. The requests are then executed using synchronous or asynchronous dedicated methods like `client.index()` or `client.indexAsync()`.
- does not provide indices or cluster management APIs. Management operations can be executed by external scripts or using the low-level client.

How to migrate the way requests are built

The Java API provides two ways to build a request: by using the request's constructor or by using a request builder. Migrating from the `TransportClient` to the high-level client can be straightforward if application's code uses the former, while changing usages of the latter can require more work.

With request constructors

When request constructors are used, like in the following example:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation
/MigrationDocumentationIT.java[migration-request-ctor]
```

① Create an `IndexRequest` using its constructor

The migration is very simple. The execution using the `TransportClient`:

```
IndexResponse response = transportClient.index(indexRequest).actionGet();
```

Can be easily replaced to use the `RestHighLevelClient`:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation
/MigrationDocumentationIT.java[migration-request-ctor-execution]
```

With request builders

The Java API provides a request builder for every type of request. They are exposed by the `TransportClient` through the many `prepare()` methods. Here are some examples:

```
IndexRequestBuilder indexRequestBuilder   = transportClient.prepareIndex();  ①
DeleteRequestBuilder deleteRequestBuilder = transportClient.prepareDelete(); ②
SearchRequestBuilder searchRequestBuilder = transportClient.prepareSearch(); ③
```

① Create a `IndexRequestBuilder` using the `prepareIndex()` method from the `TransportClient`. The request builder encapsulates the `IndexRequest` to be executed.

② Create a `DeleteRequestBuilder` using the `prepareDelete()` method from the `TransportClient`. The request builder encapsulates the `DeleteRequest` to be executed.

③ Create a `SearchRequestBuilder` using the `prepareSearch()` method from the `TransportClient`. The request builder encapsulates the `SearchRequest` to be executed.

Since the Java High Level REST Client does not support request builders, applications that use them must be changed to use requests constructors instead.

NOTE	While you are incrementally migrating your application and you have both the transport client and the high level client available you can always get the `Request` object from the `Builder` object by calling `Builder.request()`. We do not advise continuing to depend on the builders in the long run but it should be possible to use them during the transition from the transport client to the high level rest client.

How to migrate the way requests are executed

The `TransportClient` allows to execute requests in both synchronous and asynchronous ways. This is also possible using the high-level client.

Synchronous execution

The following example shows how a `DeleteRequest` can be synchronously executed using the `TransportClient`:

```
DeleteRequest request = new DeleteRequest("index", "doc", "id"); ①
DeleteResponse response = transportClient.delete(request).actionGet(); ②
```

① Create the `DeleteRequest` using its constructor

② Execute the `DeleteRequest`. The `actionGet()` method blocks until a response is returned by the cluster.

The same request synchronously executed using the high-level client is:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation
/MigrationDocumentationIT.java[migration-request-sync-execution]
```

① Execute the `DeleteRequest`. The `delete()` method blocks until a response is returned by the cluster.

Asynchronous execution

The following example shows how a `DeleteRequest` can be asynchronously executed using the `TransportClient`:

```
DeleteRequest request = new DeleteRequest("index", "doc", "id"); ①
transportClient.delete(request, new ActionListener<DeleteResponse>() { ②
    @Override
    public void onResponse(DeleteResponse deleteResponse) {
③
    }

    @Override
    public void onFailure(Exception e) {
④
    }
});
```

① Create the `DeleteRequest` using its constructor

② Execute the `DeleteRequest` by passing the request and a `ActionListener` that gets called on execution completion or failure. This method does not block and returns immediately.

③ The `onResponse()` method is called when the response is returned by the cluster.

④ The `onFailure()` method is called when an error occurs during the execution of the request.

The same request asynchronously executed using the high-level client is:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-
high-level/src/test/java/org/elasticsearch/client/documentation
/MigrationDocumentationIT.java[migration-request-async-execution]
```

① Create the `DeleteRequest` using its constructor

② Execute the `DeleteRequest` by passing the request and a `ActionListener` that gets called on execution completion or failure. This method does not block and returns immediately.

③ The `onResponse()` method is called when the response is returned by the cluster.

④ The `onFailure()` method is called when an error occurs during the execution of the request.

Manage Indices using the Low-Level REST Client

The low-level client is able to execute any kind of HTTP requests, and can therefore be used to call the APIs that are not yet supported by the high level client.

For example, creating a new index with the `TransportClient` may look like this:

```
Settings settings = Settings.builder() ①
                                .put(SETTING_NUMBER_OF_SHARDS, 1)
                                .put(SETTING_NUMBER_OF_REPLICAS, 0)
                                .build();

String mappings = XContentFactory.jsonBuilder()  ②
                                .startObject()
                                    .startObject("doc")
                                        .startObject("properties")
                                            .startObject("time")
                                                .field("type", "date")
                                            .endObject()
                                        .endObject()
                                    .endObject()
                                .endObject()
                                .string();

CreateIndexResponse response = transportClient.admin().indices()  ③
        .prepareCreate("my-index")
        .setSettings(indexSettings)
        .addMapping("doc", docMapping, XContentType.JSON)
        .get();

if (response.isAcknowledged() == false) {
④
}
```

① Define the settings of the index

② Define the mapping for document of type `doc` using a `XContentBuilder`

③ Create the index with the previous settings and mapping using the `prepareCreate()` method. The

execution is synchronous and blocks on the `get()` method until the remote cluster returns a response.

④ Handle the situation where the index has not been created

The same operation executed with the low-level client could be:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/MigrationDocumentationIT.java[migration-create-index]
```

① Define the settings of the index

② Define the body of the HTTP request using a `XContentBuilder` with JSON format

③ Include the settings in the request body

④ Include the mappings in the request body

⑤ Convert the request body from `String` to a `HttpEntity` and set its content type (here, JSON)

⑥ Execute the request using the low-level client. The execution is synchronous and blocks on the `performRequest()` method until the remote cluster returns a response. The low-level client can be retrieved from an existing `RestHighLevelClient` instance through the `getLowLevelClient` getter method.

⑦ Handle the situation where the index has not been created

Checking Cluster Health using the Low-Level REST Client

Another common need is to check the cluster's health using the Cluster API. With the `TransportClient` it can be done this way:

```
ClusterHealthResponse response = client.admin().cluster().prepareHealth().get(); ①

ClusterHealthStatus healthStatus = response.getStatus(); ②
if (healthStatus != ClusterHealthStatus.GREEN) {
③
}
```

① Execute a `ClusterHealth` with default parameters

② Retrieve the cluster's health status from the response

③ Handle the situation where the cluster's health is not green

With the low-level client, the code can be changed to:

```
include-tagged::D:/Learning/Code/elasticsearch-6.1.2/docs/java-rest/../../client/rest-high-level/src/test/java/org/elasticsearch/client/documentation/MigrationDocumentationIT.java[migration-cluster-health]
```

① Call the cluster's health REST endpoint and wait for the cluster health to become green, then get

back a `Response` object.

② Retrieve an `InputStream` object in order to read the response's content

③ Parse the response's content using Elasticsearch's helper class `XContentHelper`. This helper requires the content type of the response to be passed as an argument and returns a `Map` of objects. Values in the map can be of any type, including inner `Map` that are used to represent the JSON object hierarchy.

④ Retrieve the value of the `status` field in the response map, casts it as a a `String` object and use the `ClusterHealthStatus.fromString()` method to convert it as a `ClusterHealthStatus` object. This method throws an exception if the value does not corresponds to a valid cluster health status.

⑤ Handle the situation where the cluster's health is not green

Note that for convenience this example uses Elasticsearch's helpers to parse the JSON response body, but any other JSON parser could have been use instead.

Provide feedback

We love to hear from you! Please give us your feedback about your migration experience and how to improve the Java High Level Rest Client on our forum.

License