8-06-2020

Now today we will discuss all important classes and methods of HTTP/2 client.

These are three classes ——

HttpClient -
HttpRequest -
HttpResponse

Steps to creating HttpRequest and Process HttpResponse from Java application —

① Create HttpClient object —
② Create HttpRequest object —
③ By using HttpClient object, send request and GET HttpResponse
④ Process HttpResponse.

Steps by Coding :—

① HttpClient client = HttpClient.newHttpClient();

② String url = "https://www.google.co.in";
HttpRequest req = HttpRequest.newBuilder(new URI(url)).GET().build();

Since, newBuilder method returns Builder object hence in order to convert it to HttpRequest :—
we have to call .GET().build();

Let it make more clear as—

HttpRequest req = HttpRequest.newBuilder(new URI(url)).GET().build();

GET() method sets the request method of this builder to GET.
build() method builds and returns a HttpRequest.

## Syntax of newBuilder(), GET(), and build()

→ public static HttpRequest.Builder newBuilder(URI uri);

→ public static HttpRequest.Builder GET();

→ public abstract HttpRequest build();

③

```
HttpResponse resp = client.send(req, HttpResponse.
                    BodyHandlers.ofString());
```

or

```
HttpResponse resp = client.send(req, HttpResponse.
                    BodyHandlers.ofFile(Paths.get("abc.html")));
```

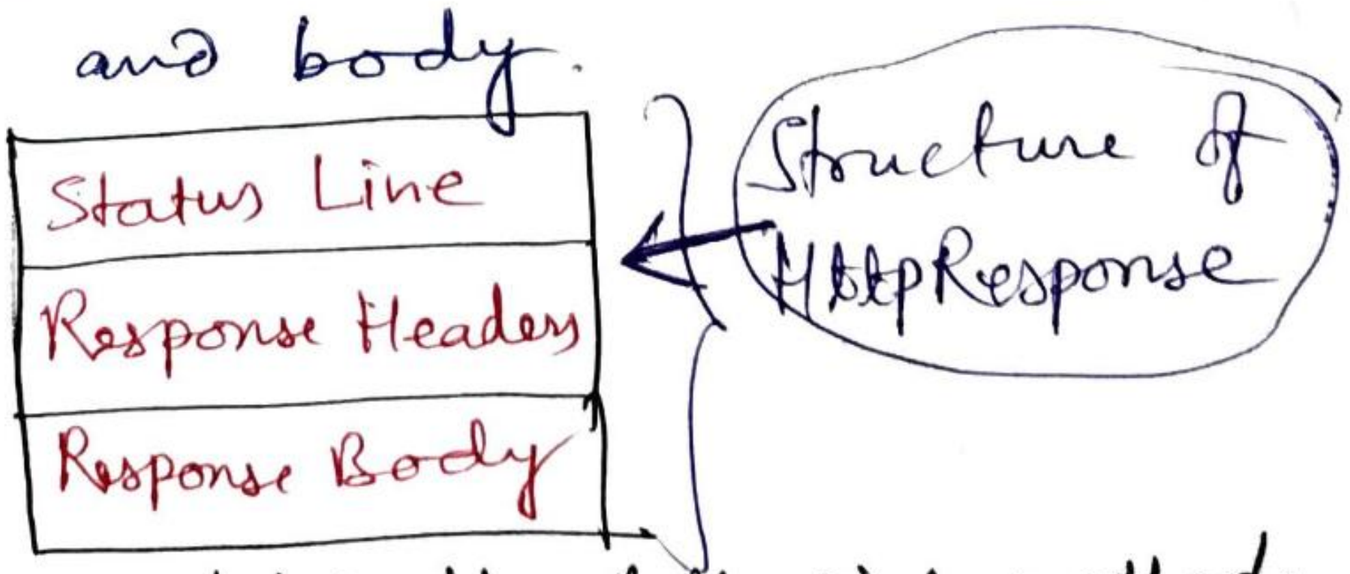
HttpClient contains the following methods :—

① send() — to send synchronous request (blocking mode)

② sendAsync() — to send asynchronous request (non blocking mode).

BodyHandler is a functional interface present inside HttpResponse. It can be used to handle body of HttpResponse.

④ → HttpResponse contains the status code, response headers and body.




HttpResponse class contains the following methods to retrieve data from the response —

① statusCode()
returns StatusCode—
(1XX, 2XX, 3XX, 4XX, 5XX)

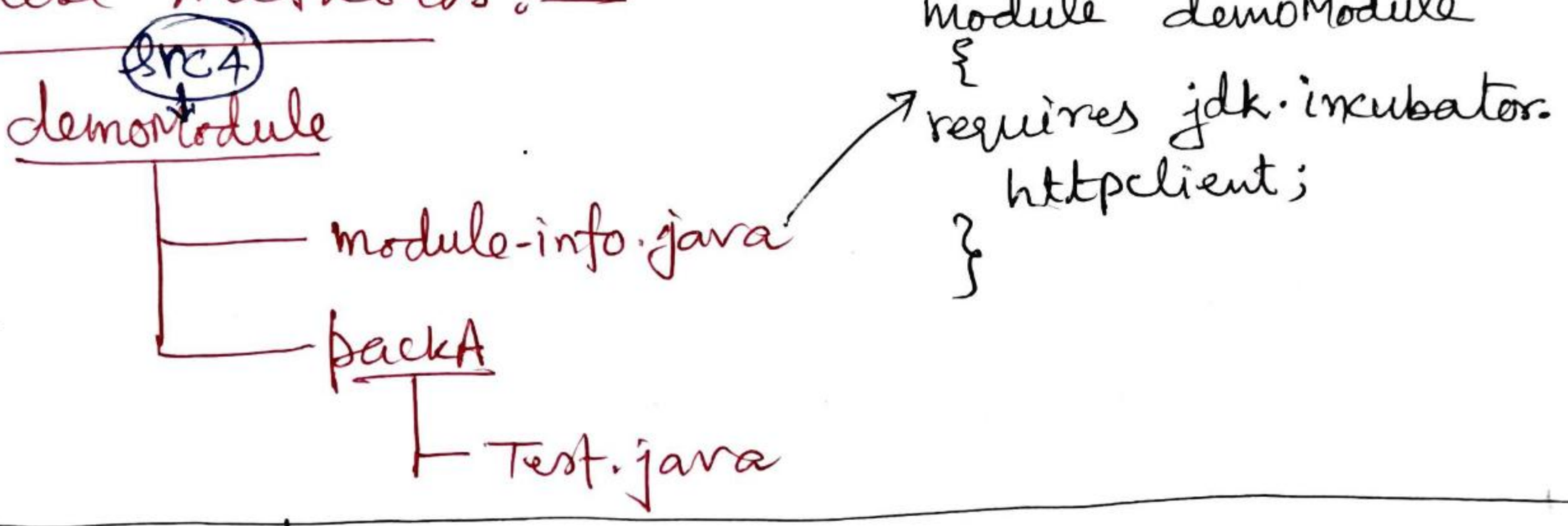
② body()
returns body of the response

③ headers()
returns header information of response.

Status code details are → with meanings.

1xx → Informational
2xx → Successful
3xx → Redirectional
4xx → Client side Error
5xx → Server side Error

Let us take a small demo program to see all these methods :—




Test.java

```
package packA; import jdk.incubator.http.*; import java.net.URI; import java.util.*;
public class Test {
    public static void main(String [] args) throws
                                                Exception
    {
    HttpClient client = HttpClient.newHttpClient();
    String url = "https://www.google.co.in";
HttpRequest req = HttpRequest.newBuilder(new URI(url)).GET().
                                                build();
HttpResponse resp = client.send(req, HttpResponse.
                    BodyHandler.asString());
System.out.println("Status Code :" + resp.StatusCode());
//System.out.println("Response Body :" + resp.body());
```

```
HttpHeaders header = resp.headers();
Map<String, List<String>> map = header.map();
System.out.println("Response Headers:");
map.forEach((k,v) -> System.out.println("\t"+ k+" :"
                                        + v));
  }
}
```

Compile :-

```
javac --module-source-path Src4 -d out4 -m
                                    demoModule ↵
```

Run :-

```
java --module-path out4 -m demoModule/packA.Test ↵
```

Status code - 200 (Successful)

→ Warning we will get because - incubator module is till under-development phase so be take care while using incubator module.

Body of a page is a big amount of HTML coding so I have commented in the program above you can uncomment to see the output.

Now, let us modify this program to get body response and store it to a file "abc.html" :-

```
import java.nio.file.Paths;
```

← Insert one line above the code after package statement.

```
HttpResponse resp = client.send(req, HttpResponse.BodyHandlers.
        asFile(Paths.get("abc.html")));
```

Paths is a class present in java.nio.file package, hence we have imported it. Hence modified program is :-

Test.java

```
package packA; import jdk.incubator.http.*;
import java.net.URI;
import java.util.List;
import java.nio.file.Paths;
public class Test{
    public static void main(String [] args) throws Exception
    {
HttpClient client = HttpClient.newHttpClient();
String url = "https://www.rediffmail.com";
HttpRequest req = HttpRequest.newBuilder(new URI(url)).
                        GET().build();
HttpResponse resp = client.send(req, HttpResponse.
            BodyHandler.asFile(Paths.get("abc.html")));
System.out.println("Status Code:" + resp.statusCode());
HttpHeaders header = resp.headers();
Map<String, List<String>> map = header.map();
System.out.println("Response Headers");
map.forEach((k,v) -> System.out.println("\t" + k +
                        ":" +v));
    }
}
```

Compile & Run Again :—

```
javac --module-source-path src4 -d out4 -m
                                    demoModule ↵
java --module-path out4 -m demoModule/packA.Test ↵
```

You have (abc.html) file in your current working directory. You can check it, open & see the result.