Http Snippet

Table of Contents

1. Usage 1
2. Har Request 3
3. Query Strings 4
4. Headers 4
5. Generators 4
6. PostData 4
    application/x-www-form-urlencoded 5
    application/json 5
    multipart/form-data 5
7. Supported Clients 6

An HTTP Request snippet generator for many languages & tools. It can generate code for over twelve different languages and currently supports cURL, Javascript, Node, C, Java, Objective-C, Swift, Python, Ruby, C#, Go, OCaml and more!.

1. Usage

Enable maven snapshots in `~/.m2/settings.xml`

```
<profiles>
  <profile>
     <id>allow-snapshots</id>
     <activation><activeByDefault>true</activeByDefault></activation>
     <repositories>
       <repository>
         <id>snapshots-repo</id>
         <url>https://oss.sonatype.org/content/repositories/snapshots</url>
         <releases><enabled>false</enabled></releases>
         <snapshots><enabled>true</enabled></snapshots>
       </repository>
     </repositories>
   </profile>
</profiles>
```

Then add this `dependency` to `pom.xml`

```
<dependency>
     <groupId>io.github.atkawa7</groupId>
     <artifactId>httpsnippet</artifactId>
     <version>0.0.1-SNAPSHOT</version>
</dependency>
```

Once you have added this as a dependency then you are ready to generate code snippets.

```
public class Main {
     public static void main(String[] args) throws Exception {
         List<HarHeader> headers = new ArrayList<>();
         List<HarQueryString> queryStrings = new ArrayList<>();

         User user = new User();
         Faker faker = new Faker();
         user.setFirstName(faker.name().firstName());
         user.setLastName(faker.name().lastName());


         HarPostData harPostData =
                 new HarPostDataBuilder()
                         .withMimeType(MediaType.APPLICATION_JSON)
                         .withText(ObjectUtils.writeValueAsString(user)).build();

         HarRequest harRequest =
                 new HarRequestBuilder()
                         .withMethod(HttpMethod.GET.toString())
                         .withUrl("http://localhost:5000/users")
                         .withHeaders(headers)
                         .withQueryString(queryStrings)
                         .withHttpVersion(HttpVersion.HTTP_1_1.toString())
                         .withPostData(harPostData)
                         .build();

         //Using default generator for the language
         HttpSnippet httpSnippet = new HttpSnippetCodeGenerator().snippet(harRequest,
Language.JAVA);
         System.out.println(httpSnippet.getCode());

         //Or directly using the generator
         String code   = new OkHttp().code(harRequest);
         System.out.println(code);

     }

     @Data
     static class User {
         private String firstName;
         private String lastName;
     }
 }
```

For integrating with other third party i.e `spring`, `redoc`, `swagger` such as checkout the demo.

```
https://github.com/atkawa7/httpsnippet
cd httpsnippet
mvn clean install
java -jar httpsnippet-demo/target/httpsnippet-demo-0.0.1-SNAPSHOT.jar
```

2. Har Request

Http snippet depends on the har request to generate code request snippets. The following object contains detailed info about performed request. Some additional fields from the Har spec are not required i.e `bodySize`, `headerSize` and `comments`.

```
{
    "method": "GET",
    "url": "http://www.example.com/path/?param=value",
    "httpVersion": "HTTP/1.1",
    "cookies": [],
    "headers": [],
    "queryString" : [],
    "postData" : {}
}
```

The following table shows mandatory and optional har request fields

Field	Type	Description	Required
httpVersion	string	Request HTTP Version.	optional
method	string	Request method (GET, POST, ...).	optional
url	string	Absolute URL of the request (fragments are not included)	required
cookies	array	List of cookie objects	optional
headers	array	List of header objects.	optional
queryString	array	List of query parameter objects.	optional
postData	object	Posted data info.	optional

The following snippet shows how to convert a json string to `HarRequest` using `jackson` object mapper.

```
  private static final ObjectMapper mapper  = new ObjectMapper();
  public static HarRequest harRequest(String jsonRequest) throws Exception{
        return mapper.readValue(jsonRequest, HarRequest.class);
  }
```

Another alternative way to create `HarRequest` is to use builders from `har-java`.

```
HarRequest harRequest =
        new HarRequestBuilder()
            .withUrl(url)
            .withQueryString(queryStrings)
            .withPostData(null)
            .build();
```

Here is how to include it as a dependency in the pom using the following

```
 <dependency>
            <groupId>com.smartbear</groupId>
            <artifactId>har-java</artifactId>
            <version>${har-java.version}</version>
        </dependency>
```

3. Query Strings

When a query string with the same name exist in both url and query string list

```
"url": "http://www.example.com/path/?foo=bar",
"queryString" : [{"name": "foo", "value": "baz"}],
```

then it will be merged into a new list and the resulting url in code snippets will be `http://www.example.com/path/?foo=bar&foo=baz`. **Note:** some servers will treat `foo` as a list when you do this. In the case where comma separated values are required passing the url as `http://www.example.com/path/?foo=bar,baz` or query string as `"queryString" : [{"name": "foo", "value": "bar,baz"}]` should suffice

4. Headers

Note: Headers are case insensitive. They are passed as key values.

5. Generators

Please start by browsing for available generators and inspect each implementation. A generator is a simple class with a constructor that accepts two parameters: language and client where language is the target language i.e `JAVA`, `PYTHON` and client is the target client that supports the language `OKHTTP` for `JAVA`. The generator has `generateCode` function which converts `HarRequest` to `HttpSnippet`.

[generators] | *generators.png*

6. PostData

application/x-www-form-urlencoded

If the post data is of type `application/x-www-form-urlencoded` then params should not be empty or containing filenames. If not then it will throw exceptions. **Note:** `text` is ignored when mimeType is `application/x-www-form-urlencoded` The following shows example of postData

```
{
    "mimeType": "application/x-www-form-urlencoded",
    "params": [{
        "name": "foo",
        "value": "bar"
    }, {
        "name": "foo",
        "value": "baz"
    }, {
        "name": "baz",
        "value": "abc"
    }]
}
```

application/json

This will match when postData.mimeType is one of: `application/json`, `text/json`, `text/x-json`, `application/x-json`. If the post data is `application/json` then params are ignored. **Note:** the text is validated and if not a valid JSON object an exception is thrown. The following shows example of postData

```
{
    "mimeType": "application/json",
    "params": [],
    "text": "{\"foo\": \"bar\"}"
}
```

multipart/form-data

This will match when postData.mimeType is one of: `multipart/mixed` `multipart/related` `multipart/form-data`, `multipart/alternative` will force `postData.mimeType` to `multipart/form-data`. The `postData` must have non empty `params` otherwise it would throw an error. If param with `fileName` exists then it must have `contentType` otherwise an error is also thrown

```
{
    "mimeType": "multipart/form-data",
    "params": [{
        "name": "foo",
        "value": "bar"
    },
    {
          "name": "foo",
          "fileName": "hello.txt",
          "contentType": "text/plain"
    }
 ]
}
```

7. Supported Clients

Client	Description
Clj-http	An idiomatic clojure http client wrapping the apache client.
NewRequest	Golang HTTP client request
JQuery	Perform an asynchronous HTTP (Ajax) requests with JQuery
XMLHttpRequest	W3C Standard API that provides scripted client functionality
Unirest	Lightweight HTTP Request Client Library
HTTP	Node.js native HTTP interface
Request	Simplified HTTP request client
NSURLSession	Foundation's NSURLSession request
CoHTTP	Cohttp is a very lightweight HTTP server using Lwt or Async for OCaml
HTTP v2	PHP with pecl/http v2
HTTP v1	PHP with pecl/http v1
cURL	PHP with ext-curl
Requests	Requests HTTP library
http.client	Python3 HTTP Client
cURL	cURL is a command line tool and library for transferring data with URL syntax
HTTPie	a CLI, cURL-like tool for humans
Wget	a free software package for retrieving files using HTTP, HTTPS
NSURLSession	Foundation's NSURLSession request
Net::http	Ruby HTTP client

Unirest	Lightweight HTTP Request Client Library
RestSharp	Simple REST and HTTP API Client for .NET
Libcurl	Simple REST and HTTP API Client for C
OkHttp	An HTTP Request Client Library
Invoke-WebRequest	Powershell Invoke-WebRequest client
JSoup	JSoup Java HTML Parser, with best of DOM, CSS, and jquery
Fetch API	Browser API that offers a simple interface for fetching resources