**Overview**

[*Feign*](https://github.com/OpenFeign/feign)*,*adeclarative*HTTP*client developed by Netflix.

*Feign* aims at simplifying *HTTP API* clients. Simply put, the developer needs only to declare and annotate an interface while the actual implementation will be provisioned at runtime.

**Setup**

First, we’ll create a new *Maven*project and include these dependencies:

|  |  |
| --- | --- |
|  | <dependency>      <groupId>io.github.openfeign</groupId>      <artifactId>feign-okhttp</artifactId>      <version>9.3.1</version>  </dependency>  <dependency>      <groupId>io.github.openfeign</groupId>      <artifactId>feign-gson</artifactId>      <version>9.3.1</version>  </dependency>  <dependency>      <groupId>io.github.openfeign</groupId>      <artifactId>feign-slf4j</artifactId>      <version>9.3.1</version>  </dependency>  <!-- https://mvnrepository.com/artifact/io.github.openfeign/feign-core -->  <dependency>  <groupId>io.github.openfeign</groupId>  <artifactId>feign-core</artifactId>  </dependency>  <!-- https://mvnrepository.com/artifact/io.github.openfeign/feign-hystrix -->  <dependency>  <groupId>io.github.openfeign</groupId>  <artifactId>feign-hystrix</artifactId>  <version>9.6.0</version>  </dependency> |

Besides the [*feign-core*](http://search.maven.org/#search%7Cgav%7C1%7Cg%3A%22io.github.openfeign%22%20AND%20a%3A%22feign-core%22) dependency (which is also pulled in), we’ll use a few plugins, especially: [*feign-okhttp*](http://search.maven.org/#search%7Cgav%7C1%7Cg%3A%22io.github.openfeign%22%20AND%20a%3A%22feign-okhttp%22) for internally using *Square’s*[*OkHttp*](http://square.github.io/okhttp/) client to make requests, [*feign-gson*](http://search.maven.org/#search%7Cgav%7C1%7Cg%3A%22io.github.openfeign%22%20AND%20a%3A%22feign-gson%22) for using *Google’s GSON* as *JSON* processor and [*feign-slf4j*](http://search.maven.org/#search%7Cgav%7C1%7Cg%3A%22io.github.openfeign%22%20AND%20a%3A%22feign-slf4j%22) for using the *Simple Logging Facade* to log requests.

To actually get some log output, you’ll need your favorite, *SLF4J*-supported logger implementation on your classpath.

Before we continue to create our client interface, we’ll set up a *Book* model for holding our data:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | public class Book {      private String isbn;      private String author;      private String title;      private String synopsis;      private String language;        // standard constructor, getters and setters  } |

**NOTE:**At least a “no arguments constructor” is needed by a *JSON* processor.

http://www.baeldung.com/spring-hateoas-tutorial

In fact, our *REST* provider is a [*hypermedia-driven API*](http://www.baeldung.com/spring-hateoas-tutorial)*,* so we’ll need a simple wrapper class:

|  |  |
| --- | --- |
| 1  2  3  4  5 | public class BookResource {      private Book book;        // standard constructor, getters and setters  } |

**Note:**We**‘**ll keep the *BookResource* simple because our sample *Feign* client doesn’t benefit from hypermedia features!

**Server Side**

To understand how to define a *Feign*client, we’ll first look into some of the methods and responses supported by our *REST* provider.

Let’s try it out with a simple *curl* shell command to list all books. Don’t forget to prefix your calls with *‘/api’*, which is the *servlet-context* defined in the *application.properties*:

|  |  |
| --- | --- |
| 1 | $> curl http://localhost:8081/api/books |

we will get a complete book repository represented as *JSON*:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35 | [    {      "book": {        "isbn": "1447264533",        "author": "Margaret Mitchell",        "title": "Gone with the Wind",        "synopsis": null,        "language": null      },      "links": [        {          "rel": "self",          "href": "<http://localhost:8081/api/books/1447264533>"        }      ]    },      ...      {      "book": {        "isbn": "0451524934",        "author": "George Orwell",        "title": "1984",        "synopsis": null,        "language": null      },      "links": [        {          "rel": "self",          "href": "<http://localhost:8081/api/books/0451524934>"        }      ]    }  ] |

We can also query individual *Book*resource, by appending the *ISBN*to a get request:

|  |  |
| --- | --- |
| 1 | $> curl http://localhost:8081/api/books/1447264533 |