1) REST with JAX RS

For these exercises and for the rest of the semester we will use Java's JAX-RS API to implement REST Services.

For this first example we will implement a simple API that consumes/provides quotes as sketched below:

Method	URI	
GET	`api/quote/{id}`	Returns the quote with the given id as: `{"quote" : "Quote text"}`
GET	`api/quote/random`	Returns a random quote as: `{"quote" : "Quote text"}`
POST	`api/quote`	Creates the quote supplied with the request as: `{"quote" : "Quote text"}` Response: `{"id": newId, "quote": "Quote text"}`
PUT	`api/quote/{id}`	Changes the quote with the given id to the text given with the request as: `{"quote" : "Quote text"}` Response: `{"id": newId, "quote": "Quote text"}`
DELETE	`api/quote/{id}`	Deletes the quote with the given ID Response: `{"quote" : "Quote text"}`

Server side:

1) Create a new NetBeans Web-project.
2) Include gson in the project
```
<dependency>
  <groupId>com.google.code.gson</groupId>
  <artifactId>gson</artifactId>
  <version>2.7</version>
</dependency>
```
3) Use the "New RESTful Web Services from Patterns" → Simple Root Resource to create a RESTfull resource class
   Important: For all your java REST projects, remember to include:
```
        <dependency>
             <groupId>org.glassfish.jersey.bundles</groupId>
             <artifactId>jaxrs-ri</artifactId>
             <version>2.23.2</version>
         </dependency>
```
4) Add the necessary changes to change the URI into "api/quote".
5) For this first example we will use a dummy in-memory data model to hold data on the server.
   Implement this map in the beginning of the Resource class
```
private static Map<Integer,String> quotes = new HashMap() {
  {
   put(1, "Friends are kisses blown to us by angels");
   put(2, "Do not take life too seriously. You will never get out of it alive");
   put(3, "Behind every great man, is a woman rolling her eyes");
   }
};
```
6) Make sure you understand WHY the map has to be static
7) Implement the first GET method (see the hints-1 for help) and test via a Browser

8) Implement the Second GET method
9) Implement the POST method and test with Postman
10) Implement the PUT method and test with Postman
11) Implement the Delete method and test with Postman

Client Side (using the REST-API via AJAX)

12) Create a new HTML-page. Provide the page with a "Quote of today" section that should fetch a random quote when the page is loaded.
13) Add a button to page created above "New Quote" that when pressed should call code that fetches a new random quote
14) Add the necessary pages (or better page) that will allow users to *Create, Edit and Delete* Quotes using the REST-API designed above