Californium

Giacomo Tanganelli
PhD student @ University of Pisa
g.tanganelli@iet.unipi.it

Californium

- Californium is a Java CoAP library.
- Download with:
  - git clone https://github.com/eclipse/californium.core.git
- Compile using Maven:
  - mvn install

- In your VM californium has been already cloned and imported into Eclipse

Classes

- CoAPServer
  - It is the base class for all custom Servers.
- CoAPClient
  - It is the base class for all custom Clients.
- Requests
  - Provides the functionality of a CoAP request. Clients instantiate such class to issue requests to Servers.
- Response
  - Provides the functionality of a CoAP response. Servers instantiate such class in reply to requests
- Option
  - A message can have several Options with different or same option numbers. Every option is associated with a value of implicit type.

Classes (2)

- CoapResource:
  - A server hosts a tree of Resources exposed to clients.
  - Resources can be added and removed dynamically.
  - CoapResource is an element on the resource tree of a server.
  - A resource must have a unique URI.
  - A resource is able to respond to CoAP requests.

Classes (3)

- CoapExchange
  - Used to match each Request to the corresponding Response
  - CoapResources handles CoAPExchange to hide the details of the CoAP protocol
  - Expose useful methods:
    - getRequestOptions: retrieve requests option set
    - getRequestedPayload: retrieve the Payload of the request
    - advanced().getRequest(): retrieve the Request object
    - …

Create a Californium Server

- Create a new Maven project
  - Edit the pom.xml according to:
    https://iotlabunipi.github.io/oM2M_development

- Create the main class which extends the base CoapServer with the main:

```
class MyServer exthends CoapServer {
        private static void main(String args[]) { … }
}
```

Define Server Resources

```
public class CoAPResourceExample extends CoapResource {
        public CoAPResourceExample(String name) {
                super(name);
        }
        public void handleGET(CoapExchange exchange) {
                exchange.respond("hello world");
        }
        public void handlePOST(CoapExchange exchange) {
                /* your stuff */
                exchange.respond(ResponseCode.CREATED);
        }
....
}
```

Create the server

- In the main method:
  - Create the Server:

```
MyServer server = new MyServer();
```

  - Add Resources

```
serve.add(new CoAPResourceExample("hello"));
```

  - Start Server

```
server.start();
```

Exercise 1

- Create a CoAP Server with a resource which allows GET and POST.
- Add another resource on the server which read a number from a query parameter and reply back with the square of the input number.

- Hint: to get the query parameters use:
  - exchange.getRequestOptions().getUriQuery()

Create a Californium Client

- Create a CoapClient object:

```
CoapClient client = new
CoapClient("coap://127.0.0.1/hello")
```

- Issue a request:

```
CoapResponse response = client.get()
```

```
CoapResponse response = client.post("10 C°",
MediaTypeRegistry.TEXT_PLAIN)
```

Advanced Client use

- Create GET Request:

  Request req = new Request.newGet()

- Add options to request:

req.getOptions().addOption(new Option(number, value))

- Add specific option through utility methods:

  req.getOptions().addUriQuery(“number=3”)

Exercise 2

- Create a CoAP Client that issue a GET request to the server of the previous exercise.

- Modify the client to add the query parameter to retrieve the square of the parameter number.

Exercise 3

- Modify the server to reply to GET according to the Accept option in the request. (Possible values: application/xml, application/json)
- Modify the client to retrieve resource representation in json and to parse the results.
- Hint: org.json

Exercise 4

- Write a client which interacts with a server deployed in cooja with contiki.
- The server must have a resource with GET and POST method.
- The client must send request to get the value and to set the value.

- Hint: remember to use the rpl-border-router