For this activity, we will use the OM2M platform that will be run locally on your machine.
Two instances of the platform are available: an IN-CSE and an MN-CSE.
The two instances of the platform must be launched separately. Both instances will authenticate to each other automatically.

You can consult the resource tree by entering the following address in your web browser: http://127.0.0.1:8080/webpage

Launch the IPE Sample on the MN-CSE:

- Go on the console of MN-CSE
- Type the command "ss ipe"
- The result of the command shows the id, the status and the name of the plugins containing the string "ipe"
- Find the id of the plugin named "org.eclipse.om2m.ipe.sample_XXX" and type in the console "start <ID>" with <ID> the identifier of the IPE.

To perform REST queries, a graphical HTTP client is recommended for ease of use. The Postman customer is advised. https://www.getpostman.com/

A) Use redirection to query the MN-CSE

Using the configuration previously launched, we end up with 2 CSE:
- An IN-CSE listening in HTTP on port 8080
- An MN-CSE listening in HTTP on port 8282

So, if you do the following HTTP requests on Postman, you get the CSE-Base from the IN-CSE and the MN-CSE respectively:

CSE-Base of IN-CSE :

Method HTTP	GET
URI	http://127.0.0.1:8080/~/in-cse
Headers	X-M2M-Origin: admin:admin

CSE-Base of MN-CSE

Method HTTP	GET
URI	http://127.0.0.1:8282/~/mn-cse
Headers	X-M2M-Origin: admin:admin

You will find that you are using a different address because the port is not the same. The MN-CSE in some cases may not be accessible from outside or simply out of reach of the end-user network.
To overcome this, the standard provides a redirection system between the different CSE. Indeed, you only need to be able to communicate with a single node of the oneM2M network to be able to access any resource as seen in the videos.

For the previous example, you will find the CSE-Base resource of the MN-CSE by sending the query to the IN-CSE:

Method HTTP	GET
URI	http://127.0.0.1:8080/~/mn-cse
Headers	X-M2M-Origin: admin:admin

You find exactly the same information as the query previously made, but it is the IN-CSE who took care of recovering the resource for you.

B) Manipulate both addressing systems

In this activity, you will see both the oneM2M addressing systems, hierarchical and non-hierarchical. The goal is to find a resource with its attributes in two different ways.
For that, you will find the data container of the LAMP_0 on the MN-CSE in both ways.
First of all :
- Find the names that make it possible to build the hierarchical URI of the resource
- Make the request to retrieve the attributes of the resource with Postman
- Find the "ri" field for Resource Identifier that corresponds to the non-hierarchical URI
- Realize the request to retrieve the attributes of the resource with the non-hierarchical URI

You will of course find that you retrieve the same information. But how to know this identifier without having recovered the resource a first time?
This information is given by the platform at the creation of the resource, where it returns the representation of the created resource. In HTTP, this information is also in the Location header in a successful resource creation response.

To test this, create a resource (a container for example) and retrieve its identifier either from the attribute "ri" or from the header Location.

C) Negotiate message content

- Get the representation of a resource in XML then in JSON.
- Create a resource using XML and receive the result in JSON
    - Provide representation of the resource to be created

To illustrate the content negotiation, you will retrieve the same information in XML and JSON.

You will first get the CSE-Base of the IN-CSE in XML as in the previous exercise:

Method HTTP	GET
URI	http://127.0.0.1:8080/~/in-cse
Headers	X-M2M-Origin: admin:admin Accept: application/xml

You will notice a new header used, presented in the course, which is Accept. It allows you to specify the type of representation you want in return.

So you can do the same query but by asking for JSON:

Method HTTP	GET
URI	http://127.0.0.1:8080/~/in-cse
Headers	X-M2M-Origin: admin:admin Accept: application/json

You notice that you get the same information but in JSON.
We will also create an AE in JSON using the following query:

Method HTTP	POST
URI	http://127.0.0.1:8080/~/in-cse
Headers	X-M2M-Origin: admin:admin Content-Type: application/json;ty=2 Accept: application/json

Content	{
	"m2m:ae": {
	"rn": "AE_CREATED_WITH_JSON",
	"api": "app-test",
	"rr": false
	}
	}

D) Manage access rights to a container

In this activity, you will modify the access rights to the resources to define a new user, and give him specific rights.
To achieve this, you will use a simulated lamp data container. Take for example the data container of the LAMP_1 lamp, ie /mn-cse /mn-name /LAMP_1_ / DATA.

First, get the attributes of the data container with a GET request. You'll find an attribute named acpi for "Access Controll Policy Identifier" that points to the ACP that manages the resource.

Then recover this ACP thanks to the identifier found. The following table allows you to decode the rules that are given by the ACP. As a reminder, the attribute "pv" corresponds to the rules applied to the linked resources, i.e. your data container; and the "pvs" attribute makes it possible to manage access to this particular ACP. Then you add up the rights to an operation to have the rights given (only one set of operation is possible).

Opération	Code
CREATE	1
RETRIEVE	2
UPDATE	4
DELETE	8
NOTIFY	16
DISCOVERY	32

You will now add a new rule to give the rights to a new user to read and discover the resources on this container. To know that the rights of the container are applied to the Content Instances which belong to him!

For this, you will modify the ACP with the following query. Consider modifying the content to fit what you want to do with the rules and putting the identifier of the relevant ACP in the URI.

Method HTTP	PUT
URI	http://127.0.0.1:8080/VOTRE_ACP
Headers	X-M2M-Origin: admin:admin Content-Type: application/xml
Contenu	<m2m:acp xmlns:m2m="http://www.onem2m.org/xml/protocols"> <pv> <acr> <acor>…</acor> <acop>…</acop> </acr> <acr> <acor>…</acor> <acop>…</acop> </acr> </pv> </m2m:acp>

You can then pretend to be your new user by changing the http X-M2M-Origin header for the new user. You can then retrieve the container or a content instance of it, but not delete it according to the rules you have established.