SCORPIO REST API

MICHAEL MEDING* , MMEDING@OUTSMARTINC.COM

CONTENTS

ABSTRACT

This article details the usage of all the Outsmart web interface RESTful API. This API provides the functionality needed to interface with all data necessary for the front end website. This article will include details regarding the usage and implementation of all available methods supported by the Outsmart app server. For the purposes of this article "BASE-URL" should be substituted with "http://dev.outsmartinc.com/scorpio" as it will be used often as a prefix for the following URL's

1 AUTHSERVICE

1.1 ping

```
#Curl script for ping
echo Ping
```

```
curl -v -H "Accept: application/json" \
BASE-URL/auth/ping
```

OUTPUT (JSON)

```
{"status":"OK","version":1.0,"errorCode":0,"errorMsg":null,"
    fieldErrors":null,"data":null}
```

URL PATTERN

BASE-URL/auth/ping

EXPLANATION Ping is a straightforward simple request-response method. It returns a 200 OK response when called.

1.2 loginAsMember

```
#Curl script for login
customer="OutSmart Power Systems"
username="uwe"
password="ccaes1"

echo Login with DTO

printf '{"customer":"%s","username":"%s",
"password":"%s"}' "$customer" $username $password > scorpio_login.data

curl -v -H "Accept: application/json" -H "Content-Type: application/json
    "
  --cookie-jar cookie_jar.txt --cookie cookie_jar.txt \
  -X POST -d @scorpio_login.data \
  http://dev.outsmartinc.com/scorpio/ \
  auth/loginAsMember
```

OUTPUT (HEADER TEXT)

```
- Replaced cookie \newline OSSESSIONID="94a3b7feab524e9204dcb84a8076a7db
    " for domain dev.outsmartinc.com, path /, expire 1404331216 \
    linebreak
- Set-Cookie: \newline OSSESSIONID=94a3b7feab524e9204dcb84a8076a7db;
    Version=1; Path=/; Max-Age=100
```

URL PATTERN

BASE-URL/auth/loginAsMember

EXPLANATION This method is the heart of AuthServie and allows the given user to login. Customer, username and password fields must match with a corresponding record in the database. Upon sucessful login a cookie named OSESSIONID is given back as a part of the response header and must be included with all other service calls. This cookie with its corresponding hash changes after each service call and is used in tracking the session and

permissions of the user. If this cookie expires, or 15 minutes of server time elapses, then the session becomes invalidated and the user will no longer be able to make service calls without logging in again.

2 LOCATIONMANAGEMENT

2.1 getAllLocations

```
// Java test code snippet
URI uri = new URI(baseUri + "LocationManagement/getAllLocations");
System.out.println("URI: " + uri.toString());

WebTarget target = client.target(uri);

String response = target.request(MediaType.APPLICATION_JSON)
  .header("Origin", "*") // must be included
  .post(null, String.class);

// create usable JSON from String
JSONObject jo = DefaultJSONFactory.getInstance()
  .jsonObject(response);
System.out.println("data: " + jo.toString());
```

OUTPUT (JSON)


```
{"status":"OK","data":[{"postalCountry":"USA","tzOffset":-1440000
    0,"weatherStationRef":1332878257808,"postalState":"MA","
    postalZip":"02054","id":750,"customerId":13,"jsonobjectName":
    "LocationDetails","name":"Millis","acquisitionZone":"DEFAULT",
    "postalCity":"Millis","active":true,"longitude":-71.354,"
    postalStreet1":"6 Milliston Rd","latitude":42.167,"
    timeZoneName":"America/New_York","postalStreet3":null,"
    postalStreet2":null},{"postalCountry":"USA","tzOffset":-14400
    000,"weatherStationRef":1332878257808,"postalState":"MA","
    postalZip":null,"id":751,"customerId":13,"jsonobjectName":"
    LocationDetails","name":"Easton","acquisitionZone":"DEFAULT",
    "postalCity":"Easton","active":true,"longitude":-71.095,"
    postalStreet1":null,"latitude":42.088,"timeZoneName":"America
    /New_York","postalStreet3":null,"postalStreet2":null}],"
    version":1}
```


URL PATTERN

BASE-URL/LocationManagement/getAllLocations

EXPLANATION The JSONString in the above output contains all the relevant information stored in an array called ”data”. Additionaly the code snippet above was written as a part of a JUnit test that included a sample login that would yeald this data. All methods from here will assume a valid login and session cookie have been issued before the method was called. This particular method returns a list of all valid locations for a given cus-

tomer (the one you are logged in as) and includes all relevant information for that location.

2.2 getLoadDetails

```java
// Java test code snippet
URI uri = new URI(baseUri + "LocationManagement/getLoadDetails");
long now = System.currentTimeMillis();

//ARGS
LocationDTO dto = new LocationDTO();
dto.setLocationId(750);
dto.setMeasurementPointId(1398368055232L);
dto.setFrom((now - (10 * 60 * 60 * 1000)));
dto.setTo(now);
dto.setDataView("min");

WebTarget target = client.target(uri);

String response = target.request(MediaType.APPLICATION_JSON)
  .header("Origin", "*") //important
  .post(Entity.json(dto), String.class);
JSONObject jo = DefaultJSONFactory.getInstance()
.jsonObject(response);
```

OUTPUT (JSON)

```
1 {"nphases":3,"location":"Uncategorized","thisMonthUsage":0,"
    periodPeak":"746","hasBreaker":true,"peakCapacity":"0","
    endUse":"HVAC/Kitchen Hood Fan","periodTotal":"7437","
    electrical":"Panel KPL1 & 2(208)/1 KHEF1","energyMates":[{"
    mac":"0x1240003BF/C"},{"mac":"0x1240003BF/B"},{"mac":"0x12400
    03BF/A"}],"ytdUsage":0,"name":"KHEF1","loadBalance":[1.682885
    348905083E11,1.7608065969601843E11,2.1944963546792365E11],"
    breaker":"1","breakerCapacity":"15","peakCapacityTime":-1234,
    "breakerPanel":"Panel KPL1 & 2(208)"}
```

URL PATTERN

BASE-URL/LocationManagement/getLoadDetails

EXPLANATION In the code snippet above I included a DTO which is a "Data Transfer Object". This object when passed as an Entity in the request gets serialized and transfered as a JSONObject to the REST call. When recieved in the REST service the object is reconstructed and used as a plain old java object (POJO). This method call given the correct parameters (as DTO or JSON argument) will return the current load data of then given measurementPointId.

3 GENERICSMANAGEMENT

3.1 getRootTags

```
// Java test code snippet
URI uri = new URI(baseUri + "GenericsManagement/getRootTags");

//ARGS
long locationId = 202L; //existing tag
GenericsManagementDTO dto = new GenericsManagementDTO(locationId);

WebTarget target = client.target(uri);

String response = target.request(MediaType.APPLICATION_JSON)
  .header("Origin", "*") //important
  .post(Entity.json(dto), String.class);
JSONObject jo = DefaultJSONFactory.getInstance().jsonObject(response);
```

OUTPUT (JSON)

```
data: {"data":"<?xml version=\"1.0\" encoding=\"UTF-8\"
    standalone=\"yes\"?><deviceTag id=\"1380746460877\" name=\"
    device_tags\" locationId=\"202\"><deviceTag id=\"138074646088
    0\" name=\"Electrical\" locationId=\"202\"/><deviceTag id=\"1
    380746461403\" name=\"End Use\" locationId=\"202\"/><
    deviceTag id=\"1380746461481\" name=\"Location\" locationId
    =\"202\"/><\/deviceTag>"}
```

URL PATTERN

BASE-URL/GenericsManagement/getRootTags

EXPLANATION A short JSON object that only returns the base parameters of each of the devices at a given locationId.

3.2 getSubTags

```
// Java test code snippet
URI uri = new URI(baseUri + "GenericsManagement/getSubTags");

//ARGS
long locationId = 1381322867514L;
boolean includeDetails = false;
GenericsManagementDTO dto = new GenericsManagementDTO(locationId); //
    searching for location 202
dto.setIncludeDetails(includeDetails);

WebTarget target = client.target(uri);
//Get JSON string
String response = target.request(MediaType.APPLICATION_JSON)
  .header("Origin", "*") //important
  .post(Entity.json(dto), String.class);
// Make object
```

```
JSONObject jo = DefaultJSONFactory.getInstance().jsonObject(response);
```

OUTPUT

```
{"data":"<?xml version=\"1.0\" encoding=\"UTF-8\" standalone=\"
    yes\"?><deviceTag id=\"1381322867351\" displayId=\"1381322867
    514\" name=\"device_tags\" locationId=\"24\"><deviceTag id=\"
    1381322867355\" name=\"Electrical\" locationId=\"24\"><
    deviceTag id=\"1381322867356\" name=\"Panel A\" locationId=\"
    24\"><deviceTag id=\"1381322867358\" name=\"10 Furnace\"
    locationId=\"24\"/><deviceTag id=\"1381322867359\" name=\"11
    Lights - Server Room, Store Room\" locationId=\"24\"/>.....}
```

URL PATTERN

BASE-URL/GenericsManagement/getSubTags

EXPLANATION This method returns a lengthy JSONString that includes all devices and their relevant information for a given locationId.

3.3 getUncategorizedTags

```
// Java test code snippet
URI uri = new URI(baseUri + "GenericsManagement/getUncategorizedTags");

//ARGS
long locationId = 1381322867514L;
GenericsManagementDTO dto = new GenericsManagementDTO(locationId);

WebTarget target = client.target(uri);

String response = target.request(MediaType.APPLICATION_JSON)
  .header("Origin", "*") //important
  .post(Entity.json(dto), String.class);
JSONObject jo = DefaultJSONFactory.getInstance().jsonObject(response);
```

OUTPUT

```
{"data":"<?xml version=\"1.0\" encoding=\"UTF-8\" standalone=\"
    yes\"?><deviceTag id=\"0\" name=\"Uncategorized\" locationId
    =\"24\"><deviceNode idref=\"1381322858725\" name=\"0x11E00009
    A/A\" isRef=\"true\"/><deviceNode idref=\"1381322866050\"
    name=\"0x11E0000C3/A\" isRef=\"true\"/><deviceNode idref=\"13
    81322865941\" name=\"0x11E0000C7/A\" isRef=\"true\"/><
    deviceNode idref=\"1381322865840\" name=\"0x11E0000C8/A\"
    isRef=\"true\"/><deviceNode idref=\"1381322858979\" name=\"0x
    11E000131/A\" isRef=\"true\"/><deviceNode idref=\"13813228591
    06\" name=\"0x11E00019C/A\" isRef=\"true\"/><deviceNode idref
    =\"1381322856258\" name=\"0x120000000/A\" isRef=\"true\"/><
    deviceNode....}
```

URL PATTERN
BASE-URL/GenericsManagement/getUncategorizedTags

EXPLANATION Returns any device tags not specified in either getRootTags or getSubTags.

4 EQUIPMENTMANAGEMENT

4.1 fetchAllEquipmentTags

```
// Java test code snippet
URI uri = new URI(baseUri + "EquipmentManagement/fetchAllEquipmentTags");


//ARGS
long locationId = 24; // existing tag
EquipmentManagementDTO dto = new EquipmentManagementDTO();
dto.setLocationId(locationId);

WebTarget target = client.target(uri);

String response = target.request(MediaType.APPLICATION_JSON)
  .header("Origin", "*") //important
  .post(Entity.json(dto), String.class);
JSONObject jo = DefaultJSONFactory.getInstance().jsonObject(response);
```

OUTPUT (JSON)

```
1 {"data":["Furnace","Kitchen App","Lights","Office","Outlets","RTU
    "]}
```

URL PATTERN
BASE-URL/EquipmentManagement/fetchAllEquipmentTags

EXPLANATION This method returns the different tags that are attached to the equipment at a given locationId

5 ENERGYMANAGEMENT

5.1 fetchBaselineData

```
// Java test code snippet
 URI uri = new URI(baseUri + "EnergyManagement/fetchBaselineData");

long now = System.currentTimeMillis();

//ARGS
EnergyManagementDTO dto = new EnergyManagementDTO();
dto.setLocationId(750);
```

```
dto.setTagId(1398368055532L);
dto.setFrom(new Timestamp(now - (10 * 60 * 60 * 1000)));
dto.setTo(new Timestamp(now));
dto.setDataView("min");

WebTarget target = client.target(uri);

String response = target.request(MediaType.APPLICATION_JSON)
  .header("Origin", "*") //important
  .post(Entity.json(dto), String.class);
JSONObject jo = DefaultJSONFactory.getInstance().jsonObject(response);
JSONArray array = jo.getJSONArray("data");
```

OUTPUT (JSON)


```
{"category_minPeriod":"ss","category_parseDates":true,"
    category_fieldName":"x","data":[{"y01":377292.4484375,"y02":4
    38328.259375,"y00":309420.7078125,"x":1404372060000},{"y01":3
    77292.4484375,"y02":438328.259375,"y00":309420.7078125,"x":14
    04372120000},{"y01":377292.4484375,"y02":438328.259375,"y00":
    309420.7078125,"x":1404372180000},{"y01":377292.4484375,"y02":
    438328.259375,"y00":309420.7078125,"x":1404372240000},{"y01":
    377292.4484375,"y02":438328.259375,"y00":309420.7078125,"x":1
    404372300000},{"y01":377292.4484375,"y02":438328.259375,"y00":
    309420.7078125,"x":1404372360000},{"y01":377292.4484375,"y02":
    438328.259375,"y00":309420.7078125,"x":1404372420000}....}
```


URL PATTERN

BASE-URL/EnergyManagement/fetchBaselineData

EXPLANATION Fetches the baseline data of a given device tagId. A valid time range must also be given. This method may return a large set of values based on the time range.

5.2 fetchCurrentPowerData

```
// Java test code snippet
URI uri = new URI(baseUri + "EnergyManagement/fetchCurrentPowerData");

long now = System.currentTimeMillis();

//ARGS
EnergyManagementDTO dto = new EnergyManagementDTO();
dto.setLocationId(750);
dto.setTagId(1398368055532L);
dto.setFrom(new Timestamp(now - (10 * 60 * 60 * 1000)));
dto.setTo(new Timestamp(now));
dto.setDataView("min");

WebTarget target = client.target(uri);

String response = target.request(MediaType.APPLICATION_JSON)
  .header("Origin", "*")
```

```
  .post(Entity.json(dto), String.class);
JSONObject jo = DefaultJSONFactory.getInstance().jsonObject(response);
JSONArray array = jo.getJSONArray("data");
```

OUTPUT (JSON)


```
{"category_minPeriod":"ss","category_parseDates":true,"
    category_fieldName":"x","data":[{"y00":316142.0911458333,"x":
    1404372060000},{"y00":316142.0911458333,"x":1404372120000},{"
    y00":318033.68072916666,"x":1404372180000},{"y00":320159.5260
    416666,"x":1404372240000},{"y00":316298.3947916667,"x":140437
    2300000},{"y00":316549.3114583333,"x":1404372360000},{"y00":3
    16329.03489583335,"x":1404372420000},{"y00":311928.7729166667,
    "x":1404372480000},{"y00":313797.12604166666,"x":140437254000
    0},{"y00":313572.5526041667,"x":1404372600000},{"y00":313185.
    6244791667,"x":1404372660000},{"y00":317352.1666666667,"x":14
    04372720000},{"y00":324432.31614583335,"x":1404372780000}....}
```


URL PATTERN

BASE-URL/EnergyManagement/fetchCurrentPowerData

EXPLANATION This method will return the power data for the given time range of the device given. Can also be quite lengthy based on the size of the time range.

5.3 fetchEnergyData

```
// Java test code snippet
URI uri = new URI(baseUri + "EnergyManagement/fetchEnergyData");
System.out.println("URI: " + uri.toString());

long now = System.currentTimeMillis();

//ARGS
EnergyManagementDTO dto = new EnergyManagementDTO();
dto.setLocationId(750);
dto.setFrom(new Timestamp(now - (10 * 60 * 60 * 1000)));
dto.setTo(new Timestamp(now));
dto.setDataView("min");

WebTarget target = client.target(uri);

String response = target.request(MediaType.APPLICATION_JSON)
  .header("Origin", "*")
  .post(Entity.json(dto), String.class);
JSONObject jo = DefaultJSONFactory.getInstance().jsonObject(response);
JSONArray array = jo.getJSONArray("data");
```

OUTPUT (JSON)

```
{"category_minPeriod":"ss","category_parseDates":true,"
    category_fieldName":"x","data":[{"y00":316142.0911458333,"x":
    1404372060000},{"y00":316142.0911458333,"x":1404372120000},{"
    y00":318033.68072916666,"x":1404372180000},{"y00":320159.5260
    416666,"x":1404372240000},{"y00":316298.3947916667,"x":140437
    2300000},{"y00":316549.3114583333,"x":1404372360000},{"y00":3
    16329.03489583335,"x":1404372420000},{"y00":311928.7729166667,
    "x":1404372480000},{"y00":313797.12604166666,"x":140437254000
    0},{"y00":313572.5526041667,"x":1404372600000},{"y00":313185.
    6244791667,"x":1404372660000},{"y00":317352.1666666667,"x":14
    04372720000},{"y00":324432.31614583335,"x":1404372780000}...}
```


URL PATTERN

BASE-URL/EnergyManagement/fetchEnergyData

EXPLANATION Fetches the raw energy data for a locationId.

5.4 fetchTemperatureData

```
// Java test code snippet
URI uri = new URI(baseUri + "EnergyManagement/fetchTemperatureData");

long now = System.currentTimeMillis();

EnergyManagementDTO dto = new EnergyManagementDTO();
dto.setLocationId(750);
dto.setFrom(new Timestamp(now - (10 * 60 * 60 * 1000)));
dto.setTo(new Timestamp(now));
dto.setDataView("min");
dto.setTempUnit("F");

WebTarget target = client.target(uri);

String response = target.request(MediaType.APPLICATION_JSON)
  .header("Origin", "*")
  .post(Entity.json(dto), String.class);
JSONObject jo = DefaultJSONFactory.getInstance().jsonObject(response);
JSONArray array = jo.getJSONArray("data");
```

OUTPUT (JSON)


```
{"category_minPeriod":"ss","category_parseDates":true,"
    category_fieldName":"x","data":[{"y00":74.67189107076695,"x":
    1404372060000},{"y00":74.67169099666543,"x":1404372120000},{"
    y00":74.6714909225639,"x":1404372180000},{"y00":74.6712908484
    6239,"x":1404372240000},{"y00":74.67109077436086,"x":14043723
    00000},{"y00":74.67089070025935,"x":1404372360000},{"y00":74.
    67069062615784,"x":1404372420000},{"y00":74.67049055205632,"x
    ":1404372480000},{"y00":74.6702904779548,"x":1404372540000},{
    "y00":74.67009040385327,"x":1404372600000},{"y00":74.66989032
    975175,"x":1404372660000},{"y00":74.66969025565024,"x":140437
    2720000},{"y00":74.66949018154872,"x":1404372780000}...}
```

URL PATTERN
BASE-URL/EnergyManagement/fetchTemperatureData

EXPLANATION As with the previous methods this returns a large JSON-String that contains raw temperature data.

5.5 fetchUseData

```
// Java test code snippet
 URI uri = new URI(baseUri + "EnergyManagement/fetchUseData");

long now = System.currentTimeMillis();

//"from":1404100800000,"to":1404152580000,"rootTagId":"1398368055532","
    locationId":750
EnergyManagementDTO dto = new EnergyManagementDTO();
dto.setLocationId(750);
dto.setRootTagId(1398368055532L);
dto.setFrom(new Timestamp(now - (10 * 60 * 60 * 1000)));
dto.setTo(new Timestamp(now));
dto.setDataView("min");

WebTarget target = client.target(uri);

String response = target.request(MediaType.APPLICATION_JSON)
  .header("Origin", "*")
  .post(Entity.json(dto), String.class);
JSONObject jo = DefaultJSONFactory.getInstance().jsonObject(response);
JSONArray array = jo.getJSONArray("data");
```

OUTPUT (JSON)

```
1 {"displayType":"rollup","data":[{"id":1398368055569,"uom":"Wh","
    name":"Refrigeration","value":"2247339","isTag":true},{"id":1
    398368055538,"uom":"Wh","name":"HVAC","value":"609211","isTag
    ":true},{"id":1398368055550,"uom":"Wh","name":"Lighting","
    value":"451544","isTag":true},{"id":1398368055554,"uom":"Wh",
    "name":"Mixed Use","value":"303588","isTag":true},{"id":13983
    68055533,"uom":"Wh","name":"Food Prep","value":"292844","
    isTag":true},{"id":1398368055557,"uom":"Wh","name":"Other","
    value":"71193","isTag":true}]}
```

URL PATTERN
BASE-URL/EnergyManagement/fetchUseData

EXPLANATION Repetative yet? This method returns the use data for the rootTagId in a given time range.

5.6 fetchTopLoads

```
// Java test code snippet
```

```
URI uri = new URI(baseUri + "EnergyManagement/fetchTopLoads");

long now = System.currentTimeMillis();

EnergyManagementDTO dto = new EnergyManagementDTO();
dto.setLocationId(750);
dto.setTagId(1398368055532L);
dto.setFrom(new Timestamp(now - (10 * 60 * 60 * 1000)));
dto.setTo(new Timestamp(now));
dto.setDataView("min");
dto.setnLoads(2);

WebTarget target = client.target(uri);

String response = target.request(MediaType.APPLICATION_JSON)
  .header("Origin", "*")
  .post(Entity.json(dto), String.class);
JSONObject jo = DefaultJSONFactory.getInstance().jsonObject(response);
JSONArray array = jo.getJSONArray("data");
```

OUTPUT (JSON)


```
{"displayType":"rollup","data":[{"id":1398368041755,"uom":"Wh","
    name":"Rack B - (Main Switch Gear (Left+Right) - R1)","value":
    "478899","isTag":false},{"id":1398368039281,"uom":"Wh","name":
    "RTU-1 - (Main Switch Gear (Left+Right) - R2)","value":"47278
    2","isTag":false},{"id":1398368003263,"uom":"Wh","name":"
    Panel PPA - (Main Switch Gear (Left+Right) - L3)","value":"32
    5569","isTag":false},{"id":1398368030859,"uom":"Wh","name":"
    Panel NLC Lighting - (Main Switch Gear (Left+Right) - R4)","
    value":"285725","isTag":false},{"id":1398368007959,"uom":"Wh",
    "name":"Comp B6 - (Rack B, Med Temp, Glycol (480) - B6)","
    value":"280447","isTag":false},{"id":1398368015632,"uom":"Wh",
    "name":"Rack A - (Main Switch Gear (Left+Right) - R5)","value
    ":"279993","isTag":false},{"id":1398368035704,"uom":"Wh","
    name":"Comp B4 - (Rack B, Med Temp, Glycol (480) - B4)","
    value":"264469","isTag":false},{"id":1398368010776,"uom":"Wh",
    "name":"ATS - (Main Switch Gear (Left+Right) - L4)","value":"
    261880","isTag":false},{"id":1398368026413,"uom":"Wh","name":
    "Comp B5 - (Rack B, Med Temp, Glycol (480) - B5)","value":"25
    7798","isTag":false},{"id":1398368017814,"uom":"Wh","name":"
    main fan (M10) - (RTU 1(480) - F1)","value":"225856","isTag":
    false},{"id":1398368034815,"uom":"Wh","name":"Panel RF (Case
    Fans) - (208V Distribution Panel (DPA) - 7)","value":"190088",
    "isTag":false}...}
```


URL PATTERN

BASE-URL/EnergyManagement/fetchTopLoads

EXPLANATION This is a particularly time consuming REST call. It averages between 15-20 seconds because it has to crunch so much data which can clearly be seen above.