

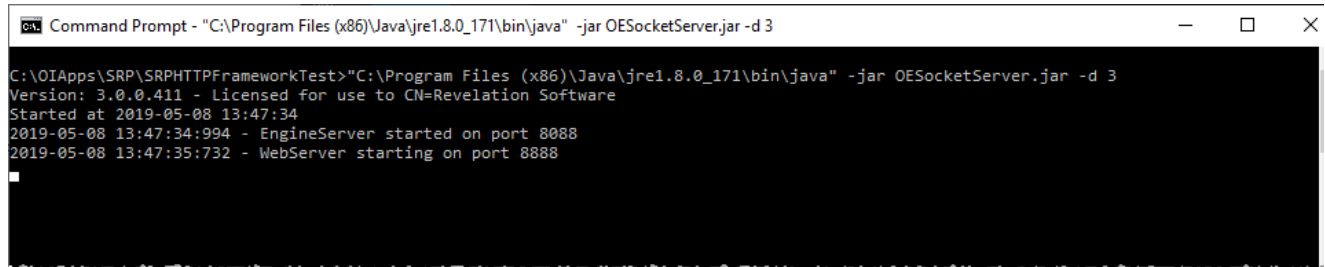
Testing for Success

Sometimes getting a taste of success is all that's needed to charge ahead and create technological magic. Therefore, before diving in and creating new APIs, let's make sure that the default APIs are responding correctly to incoming requests.

Get Your Motor Runnin'

Before we get too eager to test our APIs, let's make sure we have the Engine Server running. Otherwise our testing will fail. If using a browser, this usually results in a connection time out page. If using an API test utility like Postman, you'll see a message like "Unable to connect to Engine Server localhost: 8088 - The attempt to connect was forcefully rejected."

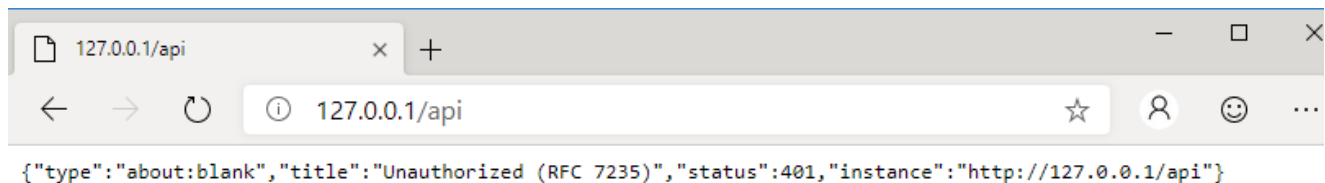
The *103-966 OpenInsight OEngineServer Configuration.pdf* document provides thorough instructions for installing the Engine Server as a true Windows service. It also provides instructions for launching it from a command prompt (aka "debug" mode). We recommend starting out by running the Engine Server from a command prompt since developers will need access to the OpenInsight Debugger when writing and troubleshooting their web APIs. Here's an example of what this should look like if everything is configured correctly:



```
Command Prompt - "C:\Program Files (x86)\Java\jre1.8.0_171\bin\java" -jar OESocketServer.jar -d 3
C:\OIApps\SRP\SRPHTTPFrameworkTest>"C:\Program Files (x86)\Java\jre1.8.0_171\bin\java" -jar OESocketServer.jar -d 3
Version: 3.0.0.411 - Licensed for use to CN=Revelation Software
Started at 2019-05-08 13:47:34
2019-05-08 13:47:34:994 - EngineServer started on port 8088
2019-05-08 13:47:35:732 - WebServer starting on port 8888
```

Knocking on the Front Door

If everything else is in good order then you are ready to run your first test. This can be done from a web browser, but unless you disabled authentication, you are likely only going to see a default 401 (i.e., unauthorized) response like this:



```
{ "type": "about:blank", "title": "Unauthorized (RFC 7235)", "status": 401, "instance": "http://127.0.0.1/api" }
```

However, this is actually good news! This confirms that the API request was received properly by the SRP HTTP Framework but it was rejected due to insufficient credentials. *(Note: older browsers such as Internet Explorer do not display the JSON response as plain text in the window as seen above. Instead, by default they will attempt to download the JSON response as a file.)*

We recommend that a good API test utility be used. This is critical for providing credentials, testing other HTTP methods (e.g., POST, PUT, PATCH, and DELETE), and sending up content for the API to process. [Postman](#) is a very popular GUI tool that is available on multiple platforms. It has a free license which will be sufficient for all your testing needs. There are other options such as a [cURL](#) and various browser plug-ins. Here's what a test of the same endpoint looks like with our pre-installed "test" user credentials:

My Workspace

Invite

Upgrade

GET 127.0.0.1/api

SRP API

Untitled Request

GET127.0.0.1/apiSendSave

ParamsAuthorizationHeaders (8)BodyPre-request ScriptTestsCookiesCodeComments (0)

TYPE

Basic Auth

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Preview Request

Username

test

Password

test

☒ Show Password

BodyCookiesHeaders (8)Test ResultsStatus: 200 OKTime: 779msSize: 909 BSave Response

PrettyRawPreviewJSON

```
1 {
2   "_links": {
3     "self": {
4       "href": "http://127.0.0.1/api"
5     },
6     "webaccounts": {
7       "href": "http://127.0.0.1/api/webaccounts",
8       "title": "Web Accounts"
9     },
10    "contacts": {
11      "href": "http://127.0.0.1/api/contacts",
12      "title": "Contacts"
13    },
14    "version": {
15      "href": "http://127.0.0.1/api/version",
16      "title": "API Version"
17    },
18    "oauth": {
19      "href": "http://127.0.0.1/api/oauth",
20      "title": "OpenAuthorization"
21    }
22  }
23 }
```

Again, if you see either of the two responses this means success! You are ready to write your own RESTful APIs.