nexl-js quick start

nexl-js is :

- REST server
- Scripting language

nexl REST server hosts JavaScripts files and exposes their primitives, arrays and objects via REST API. You might need it to store a distributed configuration/data of different servers, clients etc. in a centralized place.

nexl scripting language is intended to perform a wide variety of data manipulations with JavaScript primitives, arrays and objects hosted by nexl REST server in a single nexl expression. For example you can merge two or more JavaScript objects and produce an XML of them; or to join few arrays, eliminate duplicate elements, sort the rest elements and join them with comma.

In other words you store your configuration/data in native JavaScript form and access it via REST by using nexl scripting language which gives you a lot of power to make data manipulations on the fly.

Products and installation

There are two cross platform products : nexl-server and nexl-client.

- nexl-server is a REST server which hosts and exposes JavaScript files.
- nexl-client is a GUI application to interact with nexl-server and to simulate server's work locally.

Installation

- Download and install a latest version of nodejs
- Open a command line and write the following to install nexl-server and nexl-client:

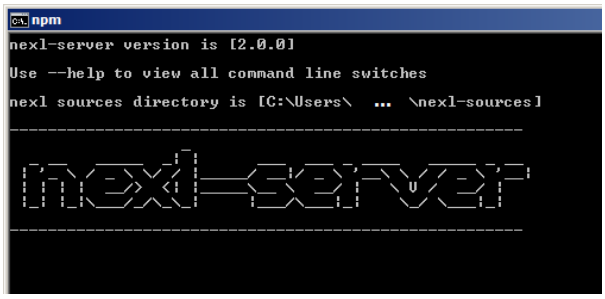
```
npm install nexl-server -g
npm install nexl-client -g
```

Creating simple JS file and exposing it via REST

Create nexl-sources directory in your ${HOME} directory (%userprofile% in Windows).
Create a example.js file with the following content and put it in the nexl-sources directory :

```
1
2  distanceToMoon = 384400;
3
4  fruits = ['Mango', 'Banana', 'Orange', 'Annona', 'Grape'];
5
6  person = {
7     name: 'Alex',
8     age: 25,
9     country: 'Canada'
10 };
11
```

Open a command line and type there nexl-server to start nexl-server



Now the example.js file is exposed via REST

You can access distanceToMoon, fruits and person variables from that file by the following URLs :

http://localhost:8080/example.js?expression=${distanceToMoon}
http://localhost:8080/example.js?expression=${fruits}
http://localhost:8080/example.js?expression=${person}

${distanceToMoon}, ${fruits} and ${person} are nexl expressions. Let's show a power of nexl expressions what they can do :

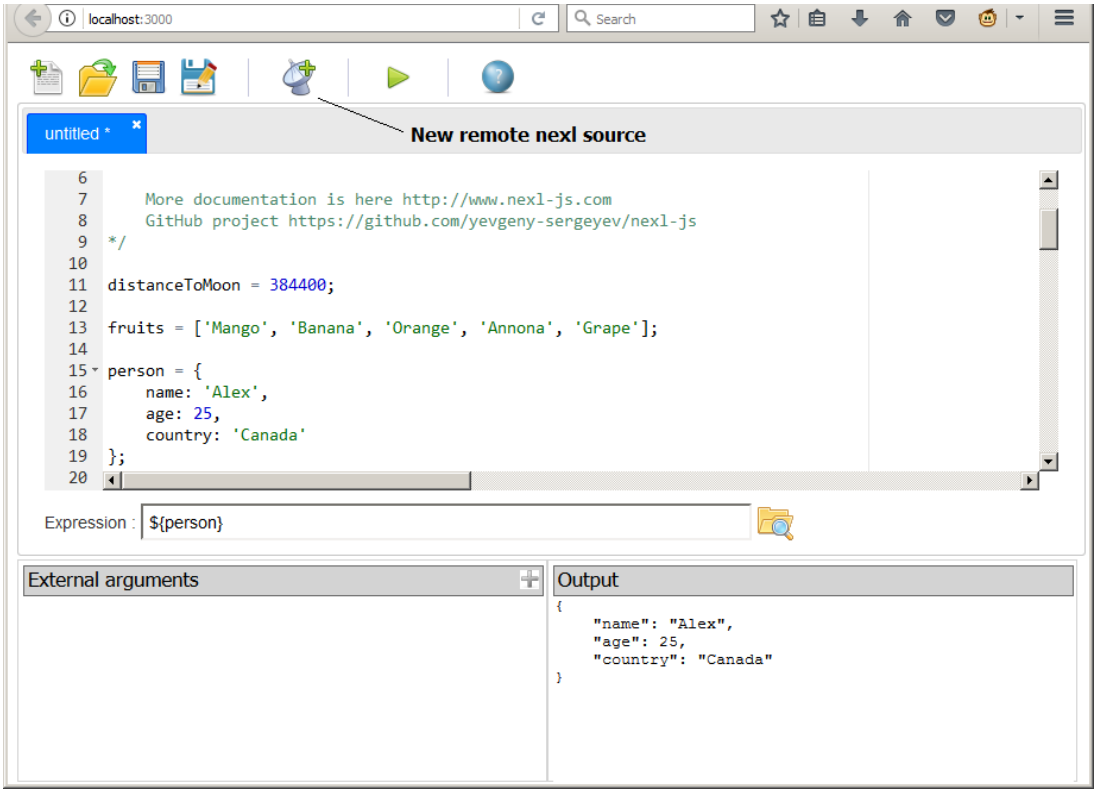
nexl expression	Explanation
${fruits&,}	Joins elements of fruits array with comma
${fruits#S}	Sorts fruits array
${fruits#S&\n}	First sorts fruits array and then joins array elements with LF
${person~X}	Produces an XML from person object

To be continued below...

nexl-client GUI application

Let's continue demonstrating nexl expressions in more convenient way by using nexl-client GUI application.

Run nexl-client by typing nexl in command line. It will open your default browser with nexl client application (it's recommended to use a Chrome or FireFox browsers).

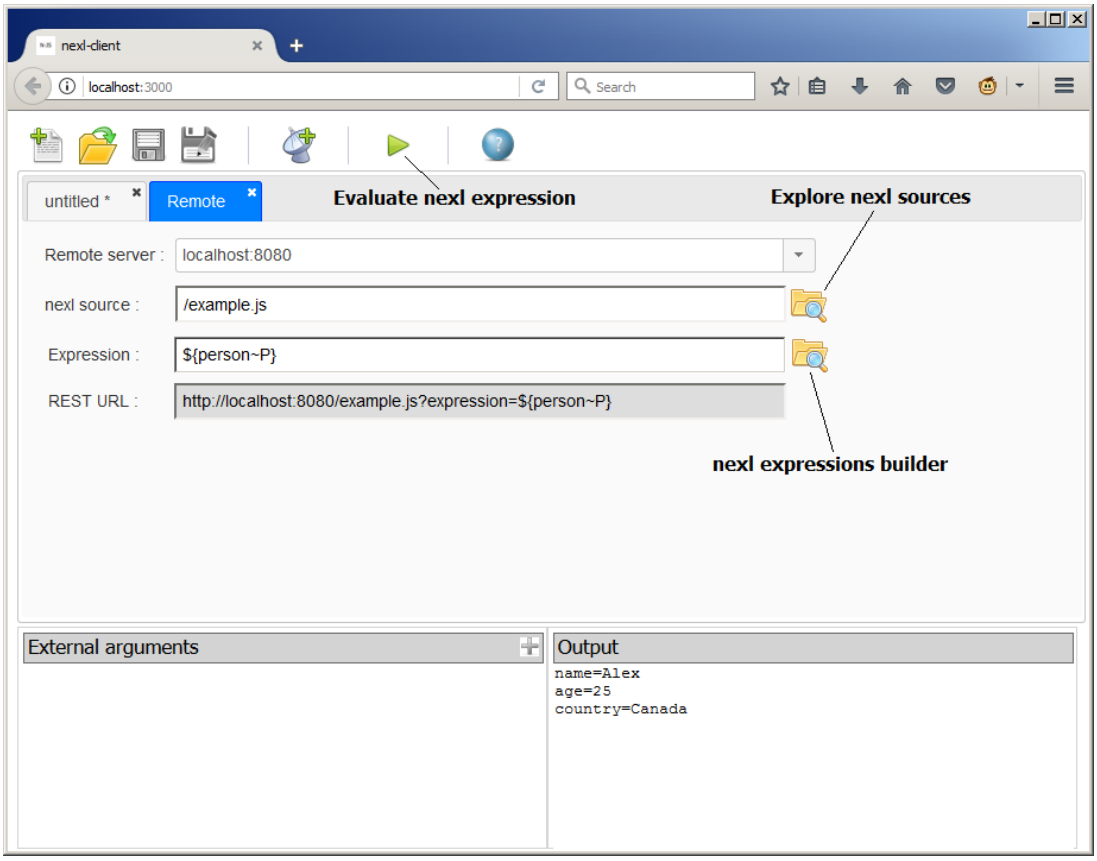


Click "New remote nexl source" button.

Enter localhost:8080 in "Remote server" field.

Enter example.js in "nexl source" field (or use explore nexl sources button)

And finally start entering nexl expressions from previous and following examples into "Expression" field to evaluate them on remote nexl-server.



Press F9 to evaluate nexl expression (or click "Evaluate nexl expression" green button)

nexl expressions examples (continued)

nexl expression	Explanation
${person.name}	First resolves person object and then resolves a name property of that object
${fruits#S}	Resolves a fruits array and sorts it

${fruits#S&,}	First sorts fruits array and then joins all elements with comma
${fruits[-1]}	Resolves a second element from the end of fruits array
${person~K}	Resolves a key set of person object as array
${person~V&,}	Resolves all values of person object as array and then joins all array elements with comma
${person~K#s[$]}	Resolves a key set of person object as array, sorts this array in descending order and finally resolves last array element
${person.country[3..$]^U1}	Resolves a country property of person object, substrings it from fourth element to the end and then capitalizes a first letter
${person~X} ${person~Y} ${person~P}	Produces an XML, YAML and key-value pairs (property file) from person object
${person<Alex}	Resolves a key of person object by 'Alex' string value (i.e. makes object property reverse resolution). The result is array
${person<Alex[0]}	Resolves a key of person object by 'Alex' value as array and then resolves a first array element
${person~K+${person~V}&\t}	Joins two arrays. The first array is a key set of a person object, the second array are values of a person object. Finally joins all array elements with tab character
${distanceToMoon~O}	Converts a distanceToMoon primitive number to JavaScript object
${distanceToMoon~O~P}	Converts a distanceToMoon primitive number to JavaScript object and then produces a key-value pair from it
${distanceToMoon~O+${person}}	Converts a distanceToMoon primitive number to JavaScript object and then merges to him person object
${Math.PI}	Resolves a PI property from Math object
${Math.PI|Math.round()}	Resolves a PI property from Math object and pushes it to stack. Calls a Math.round() function which automatically gets a Math.PI argument from the stack
${Math.PI|distanceToMoon|Math.max()}	Pushes a Math.PI to the stack, then pushes a distanceToMoon to the stack. Finally calls a Math.max() function which gets arguments from the stack

Click here for full spec and more examples of nexl scripting language