Ultimate Node.js countdown the coolest APEX examples

Alan Arentsen
Ordina
The Netherlands

Keywords:
APEX, Node.js, HTML5, Application Express

Introduction

Node.js is hot and that's not without a reason. There are numerous examples of large websites using Node.js and there is some pretty cool stuff out there.
Since APEX is officially 'awesome' and Node.js is 'hot', why not combine the two? With node-oracledb and websockets you can do awesome things, but there is more Node.js has to offer!

Extension to Apex

Node.js can be a webserver for large sites. In fact, at the moment it is serving a lot of websites. 'Wallmarkt', 'Linked In' and 'Pay-Pal' are using Node.js for their mobile websites.

In APEX, Node.js is commonly used as an extension to the functionality APEX has to offer. For example if there are some processes which you don't want to run from the database and you rather have then to run in a back end server. This can be the case with large javascript API's which every user of the web site needs to download before they can use them. Page load can be a lot faster when they are handled in a server process.

But what should APEX handle and what should be done by Node.js?
It depends…

Don't use Node.js for everything and don't use APEX for everything. Try to ask yourself the question: "What do I win?", "Will the benefits outweigh the disadvantages?"

If you need to poll on one location in the application for a process to be finished, don't use Node.js for this. But if you also need to receive live messages on screen or refresh a workload list maybe you should consider using Node.js. Especially when the application becomes very chatty in polling the database.
This is in case you are using websockets to stop the polling.

Sometimes webpages are loading so many javascript file of which they will use only 10%. If you can get the server handling these javascript functions your website may become faster.

An example of how to implement a javascript API in Node.js is the connection of a website with a social media site. Authentication will often be done with OAUTH2.
In the next example I will explain how you can integrate Dropbox in Node.js. And if you can integrate Dropbox (with OAUTH2) it should be possible to do the same with Twitter, LinkedIn and Facebook.

Configure Dropbox
Browse to the following web page to make an application in Dropbox. This application represents the container to your application private Dropbox space. File in this application get a piece of space in a Dropbox directory.

https://www.dropbox.com/developers/apps/create

After that, add a redirect URL to the application Dropbox login page. This can be any URL, for example:
http://localhost:4720/apex/5/f?p=104:11

Write down the application key and secret.

Getting the authorization codes
This example explains how to create an Dropbox connection with one fixed user. To set up this connection you will need a application key an application secret and a security token.

Open a browser window and browse to the following URL. Replace the key with your application key and the redirect URL with yours:
https://www.dropbox.com/1/oauth2/authorize?client_id=<key>&response_type=code&redirect_uri=<redirect_url>

Dropbox will ask you to allow the use of the application. Push the 'allow' button. Dropbox wil redirect you to your redirect_url. In the URL parameter section you will find a code. You will need this code to get your token.

For example:
http://localhost:4720/apex/5/f?p=104:11&code=<token>

Write down the code.

Use Curl to open the next URL. Change the key, secrets, redirect_url and code with yours:

```
curl -k https://api.dropbox.com/1/oauth2/token -d code=<code> -d
grant_type=authorization_code -d redirect_uri=<redirect_url> -u
<key>:<secret>
```

The result should be something like this:

```
{"access_token": "yourownprivatetoken"
,"token_type"  : "bearer"
,"uid"         : "123456789"}
```

You can use this token in combination with your key and secret to logon to your Dropbox application.

Write down the token:

Testing the connection
Use Curl to open the test URL. Change the token with yours:

```
curl -k https://api.dropbox.com/1/account/info -H "Authorization:
Bearer <token>"
```

result should be something like this:

```
{"referral_link" : https://db.tt/sgGeGfcx
,"display_name"  : "Alan Arentsen"
,"uid"           : 123456789
,"locale"        : "en"
,"email_verified": true
,"team"          : null
,"quota_info"    : {"datastores": 0
                   ,"shared"    : 689920629
                   ,"quota"     : 4026531840
                   ,"normal"    : 2195658875}
,"is_paired"     : false
,"country"       : "US"
,"name_details"  : {"familiar_name": "Alan"
                   ,"surname"      : "Arentsen"
                   ,"given_name"   : "Alan"}
                   ,"email"        : "alan.arentsen@gmail.com"}
```

Configure Node.js
Goto the directory in which the Dropbox API should be installed.

Install the Dropbox API with the following command:

```
npm install dropbox
```

Use the dropbox.js and dropbox_test.js files to test the connection

I hope this paper will help you start using or extending the use of Node.js next to Oracle APEX. If you want to discuss the content further, don't hesitate to contact me.

Contact address:
Alan Arentsen
Ordina
Ringwade 1
3438 MN, Nieuwegein
Phone: +31 (0) 6 13 01 88 32
Email alan.arentsen@ordina.nl
Internet: alanarentsen.blogspot.nl