Oracle APEX + Node.js

John Edward Scott
KSCOPE14
About John Scott

- Director & Founder of APEX Evangelists
- Used Oracle since version 7
- Founded APEX Evangelists (June 2007)
- Oracle Ace Director
- Author “Pro Oracle Application Express”
- Author “Expert Oracle Application Express”
- Oracle Magazine APEX Developer of the Year 2006
- “jes” on OTN Forums
A Quick Plug

All royalties go to charity
Purpose Of The Presentation

• What is Node.js
• Why it’s interesting (to APEX developers)
• How to install & use it
  • Integration points
  • Example usages
What Is Node.Js?

“Node.js is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.”

source: http://nodejs.org/
What Is Node.Js?

“Node.js is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.”

source: http://nodejs.org/
What Is Node.Js?

“Node.js is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.”

source: http://nodejs.org/
What Is Node.Js?

“Node.js is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.”

source: http://nodejs.org/
What Is Node.Js?

“Node.js is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.”

source: http://nodejs.org/
What Are Websockets?

“Javascript running on a server”

source: Me
What Are Websockets?

“Javascript running headless on a server”

source: Me
What is NodeJS?

- JavaScript runtime environment
- Runs Google Chrome V8 Engine
  - server-side JS solution
  - compiles JS = fast
- non-blocking
  - event I/O
- Designed for high-concurrency
  - event-loop rather than threading
What is NodeJS?

Traditional Synchronous method

```javascript
var data = file.read('data.csv');
process(data);
```
What is NodeJS?

Traditional Synchronous method

```javascript
var data = file.read('data.csv');
...
process(data);
```
What is NodeJS?

Traditional Synchronous method

```javascript
var data = file.read('data.csv');
......
process(data);
```
What is NodeJS?

Traditional Synchronous method

```javascript
var data = file.read('data.csv');
........
process(data);
```
What is NodeJS?

Traditional Synchronous method

```javascript
var data = file.read('data.csv');
process(data);
```
What is NodeJS?

Async method

```javascript
file.read('data.csv', function(data){
  process(data);
}
doSomethingElse();
```
Installing Node
Node.js is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Current Version: v0.10.29

AN EXAMPLE: WEBSERVER

This simple web server written in Node responds with "Hello World" for every request.

```javascript
var http = require('http');
http.createServer(function (req, res) {
    res.writeHead(200, {'Content-Type': 'text/plain'});
    res.end('Hello World');
}).listen(1337, '127.0.0.1');
console.log('Server running at http://127.0.0.1:1337/

To run the server, put the code into a file example.js and execute it with the node program from the command line:

    % node example.js
    Server running at http://127.0.0.1:1337/

Here is an example of a simple TCP server which listens on port 1337 and echoes whatever you send it:

    % node server.js
    Server running at http://127.0.0.1:1337/

    $ curl http://127.0.0.1:1337/
    Hello World
```

FEATURED

Node.js at Walmart

Eran Hammer, Principal Architect at Walmart, talks about how they use Node.js as a way to reinvent legacy APIs to create modern front end experiences.
Node.js is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Current Version: v0.10.29

AN EXAMPLE: WEBSERVER

This simple web server written in Node responds with "Hello World" for every request.

```javascript
var http = require('http');
http.createServer(function (req, res) {
    res.writeHead(200, {'Content-Type': 'text/plain'});
    res.end('Hello World');
}).listen(8888);
```

To run the server, put the code into a file example.js and execute it with the node program from the command line:

```
% node example.js
Server running at http://127.0.0.1:1337/
```

http://nodejs.org

FEATURED

reinvent legacy APIs to create modern front end experiences.

http://nodejs.org
Hello World - 2 Line Webserver

```javascript
var http = require('http');

http.createServer(function (req, res) {
    res.writeHead(200, {'Content-Type': 'text/plain'});
    res.end('Hello World
');
}).listen(1234, '127.0.0.1');

close.log('Server running at http://127.0.0.1:1234/');
```
Node Packaged Modules

Total Packages: 79 633

5 201 474 downloads in the last day
72 084 066 downloads in the last week
299 213 658 downloads in the last month

Patches welcome!

Any package can be installed by using npm install.

Add your programs to this index by using npm publish.

Recently Updated

- 2m grunt-yabs
- 4m courier.js
- 12m fossa
- 18m gulp-rev-all
- 19m stik-core.js
- 19m pinoccio
- 20m grunt-browser-sync
- 23m pdf-text-extract
- 25m astrobench
- 26m suitcss-components-flex-embed
- More...

Most Depended Upon

- 6278 underscore
- 5605 async
- 4694 request
- 3678 lodash
- 3208 commander
- 3053 express
- 2651 optimist
- 2376 coffee-script
- 2305 colors
- 1861 mkdirp
- More...
Node Packaged Modules

Total Packages: 79,633

5,201,474 downloads in the last day
72,084,066 downloads in the last week
299,213,658 downloads in the last month

Patches welcome!

Any package can be installed by using npm install.

Add your programs to this index by using npm publish.

Recently Updated

- 2m grunt-yabs
- 4m courier.js
- 12m fossa
- 18m gulp-rev-all
- 19m with-css

Most Depended Upon

- 6278 underscore
- 5605 async
- 4694 request
- 3678 lodash
- 2398 promisify

- 23m pdf-text-extract
- 25m astrobench
- 26m suitcss-components-flex-embed
- More...

- 2376 coffee-script
- 2305 colors
- 1861 mkdirp
- More...

https://www.npmjs.org/
Extensible

- NPM - Node Package Manager
- > 75,000 packages
- Very easy to install (npm install)
Using NPM

DEMONSTRATION
Using NPM

- `npm install <package>`
- `npm search <text>`
- `npm ls`
Oh the possibilities...

DEMONSTRATION
Express

DEMONSTRATION
express

web application framework for node

Web Applications
Express is a minimal and flexible node.js web application framework, providing a robust set of features for building single and multi-page, and hybrid web applications.

APIs
With a myriad of HTTP utility methods and Connect middleware at your disposal, creating a robust user-friendly API is quick and easy.

Performance
Express provides a thin layer of features fundamental to any web application, without obscuring features that you know and love in node.js

Get the book!
Check out the Manning Node.js in Action book with in-depth Connect and Express coverage, testing, databases, conventions and much more!
express
web application framework for node

http://expressjs.com/

**Web Applications**
Express is a minimal and flexible node.js web application framework, providing a robust set of features for building single and multi-page, and hybrid web applications.

**APIs**
With a myriad of HTTP utility methods and Connect middleware at your disposal, creating a robust user-friendly API is quick and easy.

**Performance**
Express provides a thin layer of features fundamental to any web application, without obscuring features that you know and love in node.js

**Get the book!**
Check out the Manning *Node.js in Action* book with in-depth Connect and Express coverage, testing, databases, conventions and much more!
Express

• Web Application framework for node
• Handles redirects / static files etc
• small footprint
A full-featured http proxy for node.js http://github.com/nodejitsu/node-http-proxy

864 commits  12 branches  50 releases  89 contributors

Merge pull request #658 from RushPL/master

jcruggz authored 15 days ago

- benchmark [dist doc] Added documentation for consistent benchmarking of node-http-proxy 9 months ago
- doc [fix] new logo 9 months ago
- examples [tests] Added a test case for run all the examples 6 months ago
- lib Change name targetTimeout to proxyTimeout 15 days ago
- test Change name targetTimeout to proxyTimeout 15 days ago
- .gitignore [fix] add 0.10 compatibility. closes #474 9 months ago
- .travis.yml Set travis to run `npm test` while we fix coveralls.io integration 6 months ago
- CHANGELOG.md fix docs 6 months ago
- LICENSE [misc] add a LICENSE file 7 months ago
- README.md [doc] update docs with toProxy option 3 months ago
- UPGRADING.md Update UPGRADING.md 3 months ago
- index.js [refactor minor] s/caronte/http-proxy/ or s/caronte/httpProxy/ where possible 9 months ago
- package.json [dist] Version bump. 1.1.4 2 months ago
A full-featured http proxy for node.js http://github.com/nodejitsu/node-http-proxy

864 commits 12 branches 50 releases 89 contributors

branch: master node-http-proxy / +

Merge pull request #658 from RushPL/master

jcrugzz authored 15 days ago

- benchmark [dist doc] Added documentation for consistent benchmarking of node-ht...
- doc [fix] new logo
- examples [tests] Added a test case for run all the examples
- lib Change name targetTimeout to proxyTimeout
- test Change name targetTimeout to proxyTimeout
- .gitignore [fix] add 0.10 compatibly.. closes #474
- .travis.yml Set travis to run `npm test` while we fix coveralls.io integration

https://github.com/nodejitsu/node-http-proxy
Http-Proxy

• Implements proxying (forward / reverse)
  • load balancing
  • traffic interception
• Supports websockets
node-oracle

DEMONSTRATION
node.js driver to connect with an oracle database.
node.js driver to connect with an oracle database.

https://github.com/joeferner/node-oracle
Publish data over REST with Node.js

Of course the best way to expose database data over REST is with Oracle REST Data Services. If you haven't read over the Statement of Direction, it's worth the couple minutes it takes. The auto table enablement and filtering is quite nice.

For anyone interested in node.js and oracle, this is a very quick example of publishing the emp table over REST for use by anyone that would prefer REST over sql*net. For example from javascript,php,perl,....

The current best driver for node.js to access oracle is node-oracle which is found here:
https://github.com/joeferner/node-oracle

Here's a short set of steps to get going.

1. Install node, http://nodejs.org/download/
2. Setup the environment's oracle variables

```bash
export PATH=$PATH:
export ORACLE_HOME=/path/to/my/oracle/product/11.2.0/dbhome_1
export OCI_LIB_DIR=$ORACLE_HOME/lib
export OCI_INCLUDE_DIR=$ORACLE_HOME/rdbms/public
export OCI_VERSION=11 #Integer. Optional, defaults to '11'
export NLS_LANG=$.UTF8 # Optional, but required to support international characters
export LD_LIBRARY_PATH=$ORACLE_HOME/lib
```

4. run npm install oracle
Publish data over REST with Node.js

Of course the best way to expose database data over REST is with Oracle REST Data Services. If you haven't read over the Statement of Direction, it's worth the couple minutes it takes. The auto table enablement and filtering is quite nice.

For anyone interested in node.js and oracle, this is a very quick example of publishing the emp table over REST for use by anyone that would prefer REST over sql*net. For example from javascript,php,perl,....

The current best driver for node.js to access oracle is node-oracle which is found here: https://github.com/joeferner/node-oracle

Here's a short set of steps to get going.

1. Install node, http://nodejs.org/download/

Basic installation

(See INSTALL.md for complete instructions for your platform.)

- Prerequisites:
  - Python 2.7 (*not* v3.x), used by node-gyp
  - C++ Compiler toolchain (GCC, Visual Studio or similar)
- Download the latest Oracle Instant Client Basic and SDK, and extract to the same directory.
- Set environment variables:

  `OCI_LIB_DIR=/path/to/instant_client`
  `OCI_INCLUDE_DIR=/path/to/instant_client/sdk/include`
  `OCI_VERSION=<10, 11, or 12> # Integer. Optional, defaults to '11'
  NLS_LANG=.$UTF8 # Optional, but required to support international characters`

- Create symlinks for libclntsh and libocci in the Instant Client directory (see INSTALL.md)
- (Linux) Install libaio
- Configure the dynamic library path on your platform to include $OCI_LIB_DIR (see INSTALL.md)
- `npm install oracle` to get the latest from npmjs.org
Basic installation

(See INSTALL.md for complete instructions for your platform.)

- Prerequisites:
  - Python 2.7 (*not* v3.x), used by node-gyp
  - C++ Compiler toolchain (GCC, Visual Studio or similar)
  - Download the latest Oracle Instant Client Basic and SDK, and extract to the same directory.

- Set environment variables:

  ```
  OCI_LIB_DIR=/path/to/instant_client
  OCI_INCLUDE_DIR=/path/to/instant_client/sdk/include
  OCI_VERSION=<10, 11, or 12> # Integer. Optional, defaults to '11'
  NLS_LANG=._UTF8 # Optional, but required to support international characters
  ```

- Create symlinks for libclntsh and liboci in the Instant Client directory (see INSTALL.md)
- (Linux) Install libaio
- Configure the dynamic library path on your platform to include $OCI_LIB_DIR (see INSTALL.md)
- `npm install oracle` to get the latest from npmjs.org
Basic installation

(See INSTALL.md for complete instructions for your platform.)

- Prerequisites:
  - Python 2.7 (not v3.x), used by node-gyp
  - C++ Compiler toolchain (GCC, Visual Studio or similar)
- Download the latest Oracle Instant Client Basic and SDK, and extract to the same directory.
- Set environment variables:

  ```
  OCI_LIB_DIR=/path/to/instant_client
 OCI_INCLUDE_DIR=/path/to/instant_client/sdk/include
  OCI_VERSION=<10, 11, or 12> # Integer. Optional, defaults to '11'
  NLS_LANG=.UTF8 # Optional, but required to support international characters
  ```

- Create symlinks for libclntsh and liboci in the Instant Client directory (see INSTALL.md)
- (Linux) Install libaio
- Configure the dynamic library path on your platform to include $OCI_LIB_DIR (see INSTALL.md)
- `npm install oracle` to get the latest from npmjs.org
Basic installation

(See INSTALL.md for complete instructions for your platform.)

- Prerequisites:
  - Python 2.7 (not v3.x), used by node-gyp
  - C++ Compiler toolchain (GCC, Visual Studio or similar)
- Download the latest Oracle Instant Client Basic and SDK, and extract to the same directory.
- Set environment variables:
  
  ```
  OCI_LIB_DIR=/path/to/instant_client
  OCI_INCLUDE_DIR=/path/to/instant_client/sdk/include
  OCI_VERSION=<10, 11, or 12> # Integer. Optional, defaults to '11'
  NLS_LANG=.UTF8 # Optional, but required to support international characters
  ```

- Create symlinks for libclntsh and libocii in the Instant Client directory (see INSTALL.md)
- (Linux) Install libaio
- Configure the dynamic library path on your platform to include $OCI_LIB_DIR (see INSTALL.md)
- `npm install oracle` to get the latest from npmjs.org
var oracle = require('oracle');

var connectData = {
    hostname: "localhost",
    port: 1521,
    database: "xe", // System ID (SID)
    user: "oracle",
    password: "oracle"
}

oracle.connect(connectData, function(err, connection) {
    if (err) {
        console.log("Error connecting to db:", err);
        return;
    }

    connection.execute("SELECT systimestamp FROM dual", [],
        function(err, results) {
            if (err) {
                console.log("Error executing query:", err);
                return;
            }

            console.log(results);
            connection.close();
        });
});
var oracle = require('oracle');

var connectData = {
    hostname: "localhost",
    port: 1521,
    database: "xe", // System ID (SID)
    user: "oracle",
    password: "oracle"
}

oracle.connect(connectData, function(err, connection) {
    if (err) { console.log("Error connecting to db:", err); return; }

    connection.execute("SELECT systimestamp FROM dual", [],
        function(err, results) {
            if (err) {
                console.log("Error executing query:", err);
                return;
            }

            console.log(results);
            connection.close();
        });
});
var oracle = require('oracle');

var connectData = {
  hostname: "localhost",
  port: 1521,
  database: "xe", // System ID (SID)
  user: "oracle",
  password: "oracle"
}

oracle.connect(connectData, function(err, connection) {
  if (err) { console.log("Error connecting to db:", err); return; }

  connection.execute("SELECT systimestamp FROM dual", [],
    function(err, results) {
      if (err) {
        console.log("Error executing query:", err);
        return;
      }

      console.log(results);
      connection.close();
    });
});
```javascript
var oracle = require('oracle');

var connectData = {
    hostname: "localhost",
    port: 1521,
    database: "xe", // System ID (SID)
    user: "oracle",
    password: "oracle"
}

oracle.connect(connectData, function(err, connection) {
    if (err) { console.log("Error connecting to db:", err); return; }

    connection.execute("SELECT systimestamp FROM dual", [],
        function(err, results) {
            if (err) {
                console.log("Error executing query:", err);
                return;
            }

            console.log(results);
            connection.close();
        });
});
```
var oracle = require('oracle');

var connectData = {
  hostname: "localhost",
  port: 1521,
  database: "xe", // System ID (SID)
  user: "oracle",
  password: "oracle"
}

oracle.connect(connectData, function(err, connection) {
  if (err) { console.log("Error connecting to db:", err); return; }

  connection.execute("SELECT systimestamp FROM dual", [],
    function(err, results) {
      if (err) {
        console.log("Error executing query:", err);
        return;
      }

      console.log(results);
      connection.close();
    });
});
Grunt

DEMONSTRATION
GRUNT
The JavaScript Task Runner

Why use a task runner?
In one word: automation. The less work you have to do when performing repetitive tasks like minification, compilation, unit testing, linting, etc, the easier your job becomes. After you've configured it, a task runner can do most of that mundane work for you—and your team—with basically zero effort.

Why use Grunt?
The Grunt ecosystem is huge and it's growing every day. With literally hundreds of plugins to choose from, you can use Grunt to automate just about anything with a minimum of effort. If someone hasn't already built what you need, authoring and publishing your own Grunt plugin to npm is a breeze.

Available Grunt plugins
Many of the tasks you need are already available as Grunt Plugins, and new plugins are published every day. While the plugin listing is more complete, here's a few you may have heard of.
GRUNT
The JavaScript Task Runner

Why use a task runner?
In one word: automation. The less work you have to do when performing repetitive tasks like minification, compilation, unit testing, linting, etc, the easier your job becomes. After you've configured it, a task runner can do most of that mundane work for you—and your team—with basically zero effort.

Why use Grunt?
The Grunt ecosystem is huge and it's growing every day. With literally hundreds of plugins to choose from, you can use Grunt to automate just about anything with a minimum of effort. If someone hasn't already built what you need, authoring and publishing your own Grunt plugin to npm is a breeze.

http://gruntjs.com/

Available Grunt plugins
Many of the tasks you need are already available as Grunt Plugins, and new plugins are published every day. While the plugin listing is more complete, here's a few you may have heard of.
Grunt

• Task Runner (built on Node.js)
  • useful for Build Automation
  • and many other things
• Vibrant Ecosystem of Plugins
  • jshint
  • uglify
  • concat
  • ...
  • extend with node
mail-listener

DEMONSTRATION
Mail listener library for node.js. Get notification when new email arrived.

Overview

Mail-listener2 library for node.js. Get notification when new email arrived to inbox. Uses IMAP protocol.

We are using these libraries: node-imap, mailparser
Mail listener library for node.js. Get notification when new email arrived.

Overview

Mail-listener2 library for node.js. Get notification when new email arrived to inbox. Uses IMAP protocol.

We are using these libraries: node-imap, mailparser
mail-listener2

• Connect to IMAP server
• Notification when email arrives
• Parse email structure
officegen

DEMONSTRATION
Node.js module: standalone Office Open XML files (Microsoft Office 2007 and later) generator for Word (docx), PowerPoint (pptx) and Excel (xlsx). The output is a stream.
Node.js module: standalone Office Open XML files (Microsoft Office 2007 and later) generator for Word (docx), PowerPoint (pptx) and Excel (xlsx). The output is a stream.
officegen

- Generate MS-Office documents
  - Excel
  - Word
  - Powerpoint
### Merge pull request #236 from adamjonas/master

**ceaess** authored on May 15

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>lib</code></td>
<td>Bump version to 0.6.2</td>
<td>3 months ago</td>
</tr>
<tr>
<td><code>spec</code></td>
<td>update rspec version and syntax from should to expect with transpec</td>
<td>a month ago</td>
</tr>
<tr>
<td><code>.document</code></td>
<td>Initial commit to PDFKit.</td>
<td>4 years ago</td>
</tr>
<tr>
<td><code>.gitignore</code></td>
<td>Merge: get first PDFKit-save-pdf test passing</td>
<td>9 months ago</td>
</tr>
<tr>
<td><code>.rspec</code></td>
<td>Actually run specs under RSpec 2</td>
<td>4 years ago</td>
</tr>
<tr>
<td><code>.ruby-gemset</code></td>
<td>Add .ruby-version, .ruby-gemset</td>
<td>9 months ago</td>
</tr>
<tr>
<td><code>.ruby-version</code></td>
<td>Add .ruby-version, .ruby-gemset</td>
<td>9 months ago</td>
</tr>
<tr>
<td><code>.travis.yml</code></td>
<td>Don't require the wkhtmltopdf gem</td>
<td>9 months ago</td>
</tr>
<tr>
<td><code>CHANGELOG.md</code></td>
<td>Bump version to 0.6.2</td>
<td>3 months ago</td>
</tr>
<tr>
<td><code>Gemfile</code></td>
<td>Stop using gemcutter as it does not support https</td>
<td>a year ago</td>
</tr>
<tr>
<td><code>Gemfile.lock</code></td>
<td>update rspec version and syntax from should to expect with transpec</td>
<td>a month ago</td>
</tr>
<tr>
<td><code>LICENSE</code></td>
<td>Initial commit to PDFKit.</td>
<td>4 years ago</td>
</tr>
<tr>
<td><code>POST_INSTALL</code></td>
<td>fixed a link to wiki in POST_INSTALL message</td>
<td>2 years ago</td>
</tr>
</tbody>
</table>
HTML+CSS to PDF using wkhtmltopdf

- 287 commits
- 4 branches
- 22 releases
- 52 contributors

**Merge pull request #236 from adamjonas/master**

**ceaess** authored on May 15

- **lib**
  - Bump version to 0.6.2
  - 3 months ago

- **spec**
  - update rspec version and syntax from should to expect with transpec
  - a month ago

- **.document**
  - Initial commit to PDFKit.
  - 4 years ago

- **.gitignore**
  - Merge: get first PDFKit-save-pdf test passing
  - 9 months ago

- **.rspec**
  - Actually run specs under RSpec 2
  - 4 years ago

- **.ruby-gemset**
  - Add .ruby-version, .ruby-gemset
  - 9 months ago

- **.ruby-version**
  - Add .ruby-version, .ruby-gemset
  - 9 months ago

**https://github.com/pdfkit/pdfkit**
pdfkit

• As it says on the tin…
  • creates PDF’s!
Enough talk...

DEMONSTRATION
Q&A

(evaluations!)
http://apex-evangelists.com