



NPM-The node package manager

Agenda

- Npm - Introduction
- Usage
 - Getting help
 - Installing package
 - Package Metadata
 - Dependencies Versioning
 -
- Important modules
 - Lodash
 - Request
 - Q
 - Async
 - Nodemailer
- Publish own package

Introduction

- npm is a **NodeJS** package manager. As its name would imply, you can use it to install node programs. Also, if you use it in development, it makes it easier to specify and link dependencies.
- npm provides a registry service for open source code to be shared and (re)used.
- Node provides the environment for running JavaScript code, but to build useful software, we need many utilities and libraries. npm is the go to place for such needs
- npm comes bundled with Node installation.

```
$: node -v && npm -v
```



Usage

- **Getting Help** npm help <term>

```
# Usage info for npm
```

```
$: npm help npm
```

```
# Manual for package.json
```

```
$: npm help json
```

```
# Help around the `init` command
```

```
$: npm init -h
```

- **Installing Packages**

- When we want to use a package, we ask npm to get it for us and make a copy on our system so that we can use it offline.
- We can have node_modules directory one per Node installation or one per working directory, commonly referred to as global install and local install respectively.

● Global Installation vs Local Installation

- Locally-If you're installing something ~~that you~~ want to use *in* your program, using `require('whatever')`, then install it locally, at the root of your project.
- If you're installing something that you want to use in your *shell*, on the command line or something, install it globally, so that its binaries end up in your `PATH` environment variable.

Syntax:

```
npm (install | i) <package(s)> (--global | -g)
```

```
npm (install | i) (--global | -g) <package(s)>
```

Install locally

```
npm install lodash --save-dev // build-time dependencies
```

```
npm install lodash --save-optional //optional dependency
```

```
npm install lodash --save //run-time dependencies
```






Package Metadata

- The package metadata that we discussed earlier is a file called `package.json` with a specified schema.
- npm will default some script values based on package contents.
 - `"start": "node server.js"`
 - `"test": "make test"`
 - `"prepublish" : "npm prune"`

Package.json

```
{  
  "name": "express",  
  "description": "Sinatra inspired web development framework",  
  "version": "2.5.11",  
  "contributors": [  
    { "name": "TJ Holowaychuk", "email": "tj@vision-media.ca" },  
  ],  
  "dependencies": {  
    "connect": "1.9.2"  
  },  
  "devDependencies": {  
    "expresso": "0.9.2",  
    "connect-redis": ">= 0.0.1"  
  },  
  "repository": "git://github.com/visionmedia/express",  
  "main": "index",  
  "scripts": {  
    "test": "make test",  
    "prepublish" : "npm prune"  
  }  
}
```



Basic commands

- **Listing Dependencies**

- `npm ls`

- **Checking Dependencies State**

- `npm outdated`

- **Updating Dependencies**

- `# To update all dependencies`

`$: npm update`

- `# To update particular dependency`

`$: npm update <package> # replace <package> with name of the package`

- **Uninstalling Dependencies**

- `# To uninstall all dependencies`

`$: npm uninstall`

- `# To uninstall particular dependency`

`$: npm uninstall <package> # replace <package> with name of the package`





node_modules



lodash

`npm install --save lodash`

Lodash makes JavaScript easier by taking the hassle out of working with arrays,

numbers, objects, strings, etc.

Lodash's modular methods are great for:

- Iterating arrays, objects, & strings
- Manipulating & testing values
- Creating composite functions



Basic functions



`_.compact`

```
_.compact([0, 1, false, 2, '', 3]);  
// => [1, 2, 3]
```

`_.concat`


```
var array = [1];  
var other = _.concat(array, 2, [3], [[4]]);  
  console.log(other);  
// => [1, 2, 3, [4]]
```

`_.indexOf`

```
_.indexOf([1, 2, 1, 2], 2);  
// => 1
```

`_.uniq`

```
_.uniq([2, 1, 2]);  
// => [2, 1]
```



__.map

```
var users = [  
  { 'user': 'barney' },  
  { 'user': 'fred' }  
];
```

```
// The `__.property` iteratee shorthand.  
__.map(users, 'user');  
// => ['barney', 'fred']
```

__.keyBy

```
__.keyBy(array, 'dir');  
// => { 'left': { 'dir': 'left', 'code': 97 }, 'right': { 'dir': 'right', 'code': 100 }  
}
```

__.now

```
console.log(__.now())
```

__.flatten

```
__.flatten([1, [2, [3, [4]], 5]]);  
// => [1, 2, [3, [4]], 5]
```

q

```
npm install q
```

A promise is an object that represents the return value or the thrown exception that the function may eventually provide.

A promise can also be used as a proxy for a remote object to overcome latency.




Pyramid of Doom

```
step1(function (value1) {  
    step2(value1, function(value2) {  
        step3(value2, function(value3) {  
            // Do something with value3  
        });  
    });  
});
```



Q.fcall

```
Q.fcall(promisedStep1)
  .then(promisedStep2)
  .then(promisedStep3)
  .then(function (value3) { // Do something with value3
    })
  .catch(function (error) { // Handle any error from all
    above steps
  })
  .done();
```

A decorative geometric pattern at the bottom of the slide, composed of various colored triangles in shades of orange, red, pink, and purple.

request

Request is designed to be the simplest way possible to make http calls.

```
var request = require('request');
request('http://www.google.com', function (error, response, body) {
  console.log('error:', error);
  console.log('statusCode:', response && response.statusCode);
  console.log('body:', body);
});

request.post({url: 'http://service.com/upload', formData: formData}, function
optionalCallback(err, httpResponse, body) {
  if (err) {
    return console.error('upload failed:', err);
  }
  console.log('Upload successful!  Server responded with:', body);
});
```



async


npm install async

- Async module provides methods to deal with Asynchronous JavaScript
- Provides common patterns for managing async code
- Can be used on client/server side




Callback Hell

```
asyncOp1(function(result) {  
  asyncOp2(function(result1) {  
    ...  
    asyncOp100(function(result99) {  
      //pew! got my result  
    });  
  });  
});  
});
```

A decorative geometric pattern at the bottom of the slide, consisting of various colored triangles and polygons in shades of orange, red, pink, and purple.


async.series

```
async.series([  
  function(cb){ ... },  
  function(cb){ // do something  
    cb(err,result);  
  }],  
  // optional callback  
  function(err,result){}  
);
```

A decorative geometric pattern at the bottom of the slide, composed of various colored triangles in shades of orange, red, pink, and purple.


async.parallel

```
async.parallel([  
  function(){ ... },  
  function(cb){ // do something  
    cb(err,result);  
  }  
],  
// optional callback  
callback);
```

A decorative geometric pattern at the bottom of the slide, composed of various colored triangles in shades of orange, red, pink, and purple.

async.waterfall

```
async.waterfall([  
  
  function(cb){cb(err,result1);},  
  
  function(arg1,cb){cb(err,result2);}  
  
],  
  
  //optional callback  
  
  callback);
```

A decorative geometric pattern at the bottom of the slide, composed of various colored triangles in shades of orange, red, pink, and purple.

Excericse

Write a function using **async.waterfall** to read data from a Json file and process on it and remove any of key and then console the new data.



Nodemailer


```
npm install nodemailer
```

- Used to send emails from node
- Can use any of the transport methods including SMTP, mailgun, SES etc.



Send using SMTP

```
var nodemailer = require('nodemailer');  
  
var transport = nodemailer.createTransport('smtp', {  
  service: 'Gmail',  
  auth: {  
    user: 'email@gmail.com',  
    pass: 'password'  
  }  
});
```

A decorative geometric pattern at the bottom of the slide, composed of various colored triangles in shades of orange, red, pink, and purple.

Sending Mail

```
var options = {from: 'Me',  
  
to: 'user_mail@gmail.com, address, list, here',  
  
subject: 'Sending Mail Using NodeMailer',  
  
html: '<b>Hey,My First Mail using NodeMailer</b>'  
  
};  
  
transport.sendMail(options, callback);
```



Exercise

1. Send a text file using nodemailer.



Publish your own Node module



Node.js modules are one kind of package which can be published to npm. When you create a new module, you want to start with the **package.json** file

Creating a user

To publish, you must have a user on the npm registry. If you don't have one, create it with **npm adduser**.

Use **npm config ls** to ensure that the credentials are stored on your client

Publishing the package

Use **npm publish** to publish the package.

Updating the package

When you make changes, you can update the package using **npm version <update_type>**, where `update_type` is one of the semantic versioning release types, patch, minor, or major. This command will change the version number in **package.json**



Exercise

Create and publish your own module



References

<https://github.com/devravitiwari/sessions-npm>

<https://lodash.com/docs/4.17.4>

<https://github.com/caolan/async/blob/v1.5.2/README.md>





Thanks.....:)

