

Nodejs web 开发简介

大纲

- nodejs 简介 (安装 ,v8, ECMA 标准兼容性 ,npm)
- nodejs web 开发框架 express 简介 (restful api, query, params, 文件上传下载, 主要结合代码实例进行介绍)
- nodejs 开发调试简介 (console.log, util.inspect, JSON.stringify, node-dev...)
- js 元编程简介 (ECMA proxy 规范, __noSuchMethod__)
- Nodejs 异步编程
- Express web 开发框架

About nodejs

<http://nodejs.org/>

- <https://github.com/joyent/node/wiki>

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.

How to install node

- Windows mac 用户直接下载 2 进制安装包即可 .
- 最新的 node 也有 linux (64 位) 二进制包 (预编译好的)
- Read the document!
- <https://github.com/joyent/node/wiki/Installation>

Http server

- A tiny http server

```
var http = require('http');
http.createServer(function(req, res) {
  console.log(req.headers);
  res.writeHead(200, {
    'Content-Type': 'text/plain'
  });
  res.end('Hello World\n');
}).listen(8080, '127.0.0.1');
console.log('Server running at http://127.0.0.1:8080/');
```

ECMA 5(6) Mozilla Features Implemented in V8

- 没有浏览器端开发要考虑那么多兼容性问题
- <https://github.com/joyent/node/wiki/ECMA-5-Mozilla-Features-Implemented-in-V8>
- <http://dailyjs.com/2012/10/15/preparing-for-esnext/>

ecma5

- `Array.forEach`
- `Object.keys`
- `Function.bind`

```
[ "a", "b", "c" ].forEach(function(item, index){
    console.log(item, index);
});

var obj = {
    'name' : 'dyw',
    'test' : function(){
        console.log(this.name);
    }
};
console.log(Object.keys(obj));

setTimeout(obj.test, 10);
setTimeout(obj.test.bind(obj), 10);
```

JSON

- JSON.stringify
- JSON.parse

```
var Tree = {  
  "name" : 'root',  
  "children" : [  
    {  
      "name" : "node-level-1"  
    },  
    {  
      "name" : "sub-node-1"  
    }  
  ]  
};  
console.log('stringified : ',JSON.stringify(Tree));  
console.log('stringified : ',JSON.stringify(Tree,null,3));  
console.log('parsed : ',JSON.parse(JSON.stringify(Tree)));
```


__defineGetter__

-

-

```
var obj = {  
  get a() {  
    return "something"  
  },  
  set a() {  
    console.log(this.a)  
  }  
};  
obj.__defineGetter__('foo', function(){  
  return 'bar'  
});  
  
obj.a = 123;  
console.log(obj.a, obj.foo);
```

node --v8-options

```
#node --v8-options > v8-options.js
```

Usage:

```
shell [options] -e string
    execute string in V8
shell [options] file1 file2 ... filek
    run JavaScript scripts in file1, file2, ..., filek
shell [options]
shell [options] --shell [file1 file2 ... filek]
    run an interactive JavaScript shell
d8 [options] file1 file2 ... filek
d8 [options]
d8 [options] --shell [file1 file2 ... filek]
    run the new debugging shell
```

Options:

```
--use_strict (enforce strict mode)
    type: bool default: false
--es5_readonly (activate correct semantics for inheriting readonliness)
    type: bool default: false
--es52_globals (activate new semantics for global var declarations)
    type: bool default: false
--harmony_typeof (enable harmony semantics for typeof)
    type: bool default: false
--harmony_scoping (enable harmony block scoping)
    type: bool default: false
--harmony_modules (enable harmony modules (implies block scoping))
    type: bool default: false
--harmony_proxies (enable harmony proxies)
    type: bool default: false
--harmony_collections (enable harmony collections (sets, maps, and weak maps))
    type: bool default: false
--harmony (enable all harmony features (except typeof))
    type: bool default: false
```

node --harmony_proxies(动态元编程)

- node --harmony or node --harmony_proxies
- http://soft.vub.ac.be/~tvcutsem/invokedynamic/proxies_tutorial

```
//v8(nodejs),firefox 支持Proxy  
//node --harmony harmony-proxy.js |  
var model = Proxy.create({  
  get: function(proxy, name) {  
    return 'Hello, ' + name;  
  }  
});  
  
console.log(model.dyw); // should print 'Hello, dyw'
```

Spider monkey __noSuchMethod__

- __noSuchMethod__
- 仅 Firefox 浏览器支持

```
var obj = {
  __noSuchMethod__: function(methodName, args) {
    console.info(methodName, args);
    obj[methodName] = function(args){//define method dynamically!
      console.warn(methodName + ' invoked with args:',args);
    };
  }
};
obj.testFunction(1, 'abc', 234); //run it in firebug!
obj.testFunction({
  "test" : 123
}); //run it in firebug!
```

node-dev

- `node test.js`
- `node-dev test.js` // 监控 js 内容变化，即时重启 node

npm

- npm publish(npm adduser)
- npm install express -g
- npm update express -g
- **package.json**

异步 vs 同步

- `fs.readFileSync('net.js', 'utf-8');`
- `fs.readFile('net.js', 'utf-8', function(err, data) {});`
- nodejs 编程推荐都用异步方式，以保证无阻塞。
-

```
var fs = require('fs');

fs.readFile('net.js', 'utf-8', function(err, data) {
  if (err) throw err;
  console.log('async: ', data);
});

console.log('Sync: ', fs.readFileSync('net.js', 'utf-8'));
```

异步回调的嵌套陷阱

- promises

```
Parse.User.logIn("user", "pass", {  
  success: function(user) {  
    query.find({  
      success: function(results) {  
        results[0].save({ key: value }, {  
          success: function(result) {  
            // the object was saved.  
          }  
        });  
      }  
    });  
  }  
});
```

```
Parse.User.logIn("user", "pass").then(function(user) {  
  return query.find();  
}).then(function(results) {  
  return results[0].save({ key: value });  
}).then(function(result) {  
  // the object was saved.  
});
```


使用 promise 解决回调嵌套

- promise

```
Parse.User.logIn("user", "pass", {
  success: function(user) {
    query.find({
      success: function(results) {
        results[0].save({ key: value }, {
          success: function(result) {
            // the object was saved.
          },
          error: function(result, error) {
            // An error occurred.
          }
        });
      },
      error: function(error) {
        // An error occurred.
      }
    });
  },
  error: function(user, error) {
    // An error occurred.
  }
});
```

```
Parse.User.logIn("user", "pass").then(function(user) {
  return query.find();
}).then(function(results) {
  return results[0].save({ key: value });
}).then(function(result) {
  // the object was saved.
}, function(error) {
  // there was some error.
});
```

jQuery Promise

- <http://api.jquery.com/jquery.ajax/>
- The jqXHR objects returned by `$.ajax()` as of jQuery 1.5 implement the Promise interface, giving them all the properties, methods, and behavior of a Promise (see **Deferred** object for more information).

```
var jqxhr = $.ajax("example.php").done(function() {  
    alert("success");  
}).fail(function() {  
    alert("error");  
}).always(function() {  
    alert("complete");  
});|
```

Express web framework

- **Sinatra inspired** web development framework for node.js -- insanely fast, flexible, and simple
- Built on ***Connect*** Middleware
- <http://expressjs.com>
- `npm install -g express`
- Read the doc!!

```
//npm install -g express
var express = require('express');
var app = express();

app.get('/', function(req, res){
  res.send('Hello World');
});

app.listen(3000);
```

What Express Does

- Parses arguments and headers
- Routing
- Views
- Partials
- Layouts
- Configuration
- Sessions

Request Object

- req.param
- req.query
- req.body
- Req.files
- req.flash ---- show error msg
- req.session , req.cookies express.cookieParser;
- req.headers

Response Object

- `res.redirect`
- `res.write` ---Transfer-Encoding:**chunked**
- `res.end`
- `res.download` `res.sendfile` `res.attachment`
- `res.send`
- `res.status`
- `res.json` `res.jsonp`
- `res.format` `res.render`

express--CRUD

- CRUD <---> HTTP Get,Post,Put,Delete
- restful

```
//npm install -g express
var express = require('express');
var app = express();

app.get('/', function(req, res){
  res.send('get!');
});
app.post('/', function(req, res){
  res.send('post!');
});
app.put('/', function(req, res){
  res.send('put!');
});
app.delete('/', function(req, res){
  res.send('delete!');
});
app.listen(3000);
```

Get-Post-Put-Delete

The screenshot displays the Postman application interface. On the left sidebar, the 'History' tab is active, showing a list of recent requests: a POST to `http://127.0.0.1:3000/upload`, a DELETE to `http://127.0.0.1:3000/`, a PUT to `http://127.0.0.1:3000/`, another POST to `http://127.0.0.1:3000/`, and a GET to `http://127.0.0.1:3000/`.

The main workspace shows the details of the selected POST request. The URL bar contains `http://127.0.0.1:3000/upload`. The 'form-data' tab is selected for the request body, with a single key-value pair: 'test' with the value 'uploaded'. The 'Send' button is highlighted in blue, indicating a successful execution. Below the request details, the response section shows a status of '200 OK' and a response time of '44 ms'. The 'Body' tab is active, displaying the response content as '1 uploaded' in a 'Pretty' format.

POSTMAN

History Collections

POST `http://127.0.0.1:3000/upload`

DELETE `http://127.0.0.1:3000/`

PUT `http://127.0.0.1:3000/`

POST `http://127.0.0.1:3000/`

GET `http://127.0.0.1:3000/`

Normal Basic Auth Digest Auth OAuth 1.0 No environment

`http://127.0.0.1:3000/upload`

form-data x-www-form-urlencoded raw

test Choose Files No file chosen File

Key Value Text

Send Preview Add to collection

Body Cookies (10) Headers (5) **STATUS** 200 OK **TIME** 44 ms

Pretty Raw Preview JSON XML

1 uploaded

express.methodOverride

- `<input type="hidden" name="_method" value="delete"/>` 附加参数 `_method`.
- `app.use(express.bodyParser());`
- `app.use(express.methodOverride());`
- `jQuery.ajax()` 通过 **type** 参数支持 http PUT 及 DELETE 方法 (部分浏览器 XMLHttpRequest 对象支持 put,delete 方法)

express—file upload

- `app.use(express.bodyParser)`
- <https://github.com/felixge/node-formidable>

```
var express = require('express'), http = require('http');

var app = express();

app.set('port', process.env.PORT || 3000);
app.use(express.bodyParser({
  keepExtensions: true,
  uploadDir: '/tmp/express/files' //mkdir -p /tmp/express/files
}));

app.post('/upload', function(req, res){
  console.log(req.files);
  res.send('uploaded');
});

http.createServer(app).listen(app.get('port'), function() {
  console.log('Express server listening on port ' + app.get('port'));
});
```