

ExpressJS

Gopalakrishnan Subramani

www.nodesen.se

gs@nodesen.se

Install

```
> npm install express --save
```

Application

```
var express = require('express');  
var app = express();
```

Express APP

```
app.get('/', function(req, res){  
  res.send('hello world');  
});
```

GET / HTTP/1.1

response: to client

request from client

```
app.listen(3000);
```

localhost:3000/

Server Port (listen)

```
app.listen(8080);
```

```
//Listen on all ethernet IP v4 addresses
```

```
app.listen(8080, "0.0.0.0");
```

```
//Listen on specific ethernet IP v4 addresses
```

```
app.listen(8080, "192.168.1.100");
```

```
//Listen on all ethernet IP v6, IP V4 addresses
```

```
app.listen(8080, "::");
```

HTTP Methods

```
app.get('/', function(req, res){  
  res.send('hello world');  
});
```

GET

```
app.post('/save', function(req, res){  
  res.send('saved');  
});
```

POST

```
app.put('/update/1', function(req, res){  
  res.send('updated');  
});
```

PUT

```
app.delete('/delete/1', function(req, res){  
  res.send('deleted');  
});
```

DELETE

Response

res.send() sends entire response to client

```
app.get("/html", function(req, res) {  
  //set content-type to text/html  
  res.send("<h2>hello</h2>");  
})
```

Response Write

```
router.get("/response/write", function(req, res) {  
  //chunk of data stream  
  res.write("<h1>hello</h1>");  
  res.write("<h1>next line</h1>");  
  //must end  
  res.end()  
})
```

Response

```
app.get("/html", function(req, res) {  
    //set content-type to text/html  
    res.send("hello");  
})
```

```
app.get("/json", function(req, res) {  
    //set content-type to application/json  
    res.send({name: "krish"});  
})
```

//or

```
app.get("/json", function(req, res) {  
    //set content-type to application/json  
    res.json({name: "krish"});  
})
```


Response with StatusCode

```
app.get("/status", function(req, res) {  
    //set content-type to application/json  
    res.status(404).send('Sorry, we cannot find that!');  
})
```

```
app.get("/status-code", function(req, res) {  
    //status with fixed/standard http reason  
    res.sendStatus(403); //Forbidden  
})
```

Response Redirect

```
app.get("/redirect", function(req, res){  
  //Send status 302  
  //& Set Location header to new url  
  res.redirect("/moved-there");  
})
```

Headers

req.headers to get request header

```
app.get("/get-headers", function(req, res){  
    res.send(req.headers['user-agent'])  
})
```

Add/Remove Header

//Set new Header in response

```
app.get("/set-header", function(req, res){  
    res.set("X-Custom", "TOKEN-1232232" );  
    res.send("set");  
})
```

//Remove Header in response

```
app.get("/headers/remove", function(req, res){  
    res.removeHeader('X-Powered-By');  
    res.send("removed");  
})
```

Query Params

```
app.get('/query', function (req, res) {  
  res.send(req.query.name + ": " + req.query.value);  
})
```

Routing

- Application End Point
- Matches Url to Controller

Methods

- `app.get()`
- `app.post()`
- `app.delete()`
- `app.put()`

all method

.all() is called for GET,POST, DELETE, PUT Methods,
next() shall make next available handler

```
app.all("/router/all", function(req, res, next){  
    res.write("<h1>From all</h1>")  
    next();  
})
```

```
app.get("/router/all", function(req, res) {  
    res.write("<h2>From get all</h2>");  
    res.end();  
})
```


Patterns

//matches acd or abcd

```
app.get('/router/pattern/ab?cd', function (req, res) {  
  res.send('ab?cd')  
})
```

//matches abcd, abbcd, abbbbbbbcd etc

```
app.get('/router/pattern/ab+cd', function (req, res) {  
  res.send('ab+cd')  
})
```

//must start with ab and end with cd

//abcd, abfd123cd

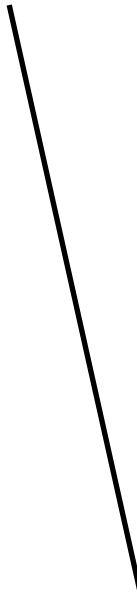
```
app.get('/router/pattern/ab*cd', function (req, res) {  
  res.send('ab*cd')  
})
```

Route Params

```
app.get('/router/product/:id/reviews/:reviewId',  
  function (req, res) {  
    console.log(req.params.id)  
    console.log(req.params.reviewId)  
    res.send(req.params)  
  })
```

Route Handler

```
function auth(req, res, next) {  
  if (req.user)  
    next();  
  
  res.sendStatus(403);  
}  
  
router.get("/router/secured", auth,  
  function(req, res) {  
    res.send("have access");  
  })
```



Router

- Create multiple files with routing, code segregation

```
//about.js
```

```
var express = require('express')
```

```
var router = express.Router()
```

```
router.get('/about', function (req, res) {  
  res.send('Node.js consultancy')  
})
```

```
router.get('/contact', function (req, res) {  
  res.send('anywhere in the web')  
})
```

```
module.exports = router
```

Using Router

```
//app.js
```

```
var about = require('./about')
```

```
// ...
```

```
app.use(about)
```

```
//http://localhost/about
```

```
//http://localhost/contact
```

```
//or
```

```
app.use("/company", about)
```

```
//http://localhost/company/about
```

```
//http://localhost/company/contact
```

Cookies

```
> npm install cookie-parser --save
```

```
var cookieParser = require("cookie-parser");
```

```
app.use(cookieParser("secret"));
```

Cookies

```
app.get('/cookies/remove', function (req, res) {  
  res.clearCookie("CookieName");  
  res.send("Cookie removed");  
})
```

Cookie - Read Values

```
app.get('/cookies/get', function (req, res) {  
    if (req.cookies["CookieName"]) {  
        res.send("cookie is " + req.cookies["CookieName"]);  
        return;  
    }  
  
    res.send('cookie not found')  
})
```


Cookies

```
app.get('/cookies/set', function (req, res) {  
  res.cookie("CookieName",  
             "Value",  
             { maxAge: 3600, httpOnly: true });  
  res.send("cookie set");  
})
```

Signed Cookie

```
router.get("/cookies/signed", function(req, res){  
  res.cookie('session', '12345456675', {signed: true})  
  res.send("Signed cookie");  
})
```

Set-Cookie:session=s
%3A12345456675.VrxMdfGxtqEAq7ix0BjURbnUtYjo7%2BvNIT9tx8aN2E;
Path=/

Read Signed Cookies

```
router.get("/cookies/read-signed", function(req, res){  
  // decoded cookie value  
  res.send("Signed " + req.signedCookies['session']);  
})
```

Remove Cookies

```
app.get('/cookies/remove', function (req, res) {  
    res.clearCookie("CookieName");  
    res.send("Cookie removed");  
})
```

Removing All Cookies

```
app.get('/logout', function (req, res) {  
  for (var cookieName in req.cookies) {  
    console.log("cleaning cookie ", cookieName);  
    res.clearCookie(cookieName);  
  }  
  res.send('removed all cookies')  
})
```

Middleware

- Middleware are functions
- Chained in between request and handlers
- Middleware can make changes to request/response
- Middleware can call next middleware function in the stack

App level middleware

```
app.use(function (req, res, next) {  
  console.log('Time Start:', Date.now())  
  next()  
  console.log('Time End:', Date.now())  
})
```

Router Level Middleware

```
var express = require("express")

var router = express.Router();

// a middleware function with no mount path.
//This code is executed for every request to the router
router.use(function (req, res, next) {
  console.log('Time:', Date.now())
  next()
})
```


Error Handling Middleware

//must takes four arguments

//useful for handling exceptions, crashes

```
app.use(function (err, req, res, next) {  
  console.error(err.stack)  
  res.status(500).send('Something broke!')  
})
```

Middlewares

- Express has more middleware libraries
- passport js, cookie parser, body parser, multer

> npm install cookie-parser

```
app.use(cookieParser())
```

Static File Serving

```
//serve the files from public folder  
app.use(express.static("public"));
```

```
//serve the files through /static url from current dir/  
public folder  
app.use('/static', express.static(path.join(__dirname,  
'public')))
```

Static Files & ETag

- “ETag” header stands for Entity Tag, often used for caching, validating/invalidating cached files
- Server send ETag header for files (ETag: ABCD123335), especially ExpressJS static file middle uses MD5 Hash for checksum.
- Browser try to check with browser if the File is changed or not by using Sending “If-None-Match: ABCD123335”
- ETag is enabled by default
- To disable cache tag, `express.static(myStaticPath, {etag: false})`

Static File-MaxAge

- To manage Cache Control header for caching
Cache-Control: public, max-age=86400

```
app.use(express.static(myStaticPath, {  
  //one day in milliseconds  
  maxage: 86400000  
}))
```

Static - Index/Default page

- When the request is made to a directory, “index” option helps to serve the default file

```
app.use(express.static(myStaticPath, {  
  index: 'index.html'  
}))
```