






This repository Search


[Pull requests](#) [Issues](#) [Gist](#)


 [cydneymikel](#) / **CS454**

 Watch

11

 Star



8


 Fork

18

Branch: **master**

CS454 / Week05 / Week5.pdf

 **cydneymikel** week 5



b964322 9 days ago







1 contributor

110 KB

Raw

History

CS454 Node.js & Angular.js

Cydney Auman
Albert Cervantes
CSULA

Combining Node & Angular - Week 5

Node Core Modules

HTTP/HTTPS

- Interfaces designed to support features of the http or https protocol. It provides functionality for running HTTP servers and making HTTP requests.

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

Node HTTP Server

A function passed as an argument to `createServer` and is called every time a client tries to connect to the server.

The `request` and `response` variables are objects representing the incoming and outgoing

data.

```
var http = require("http");

var server = http.createServer(function(request, response) {
  response.writeHead(200, { "Content-Type": "text/plain" });
  response.end("Hello World");
});

server.listen(8000, "localhost");
console.log("Server running at http://localhost:8000/");
```

Request

`request` is a request that comes from the client - this sometimes shortened to `req`.

The request argument contains information about the request, such as its url, http method, headers and etc.

```
http.createServer(function ( request, response) {
  console.log("Request URL: " + request.url);
  console.log("Request Method: " + request.method);
  console.log("Request Headers: " + JSON.stringify(request.headers));
});
```

Response

The `response` is the next argument in the function. Just like the prior argument is often shortened to `res`.

```
http.createServer(function (request, response) {

  response.writeHead(200, {"Content-Type": "text/plain"});
  response.end("Server is running.");

});
```

With each response, you get the data ready to send, and then you call `response.end()`. Eventually, you **must** call this method. This method does the actual sending of data. If this method is not called, the server just hangs forever.

Node with Express

Express.js describes itself as a "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."

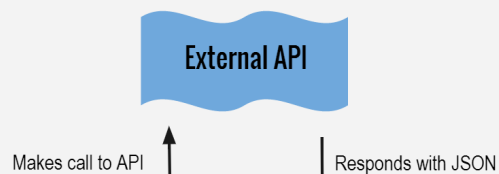
In short, it's a framework for building web applications with Node.js.

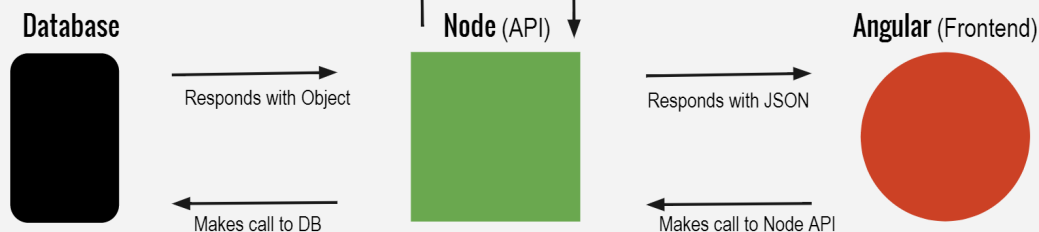
Node with Express

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

app.get("/", function(request, response) {
  response.writeHead(200, { "Content-Type": "text/plain" });
  response.end("Hello World!");
});
app.listen(8000);
```

Overall Architecture





Angular Factories

Another feature of AngularJS is the ability to encapsulate data functionality into factory, service or provider.

With the factory you create an object inside of the factory and return it.

```
angular.module('numbers.service', ['ngResource'])
  .factory('numbersResource', function($resource) {

    return {
      api: $resource('/api/:type', {}, {})
    };
  });
```

Angular Modules

ngResource

- module provides interaction support with RESTful services
- <https://ajax.googleapis.com/ajax/libs/angularjs/1.4.7/angular-resource.js>

ngRoute

- module provides routing and deeplinking services and directives for angular apps.
- <https://ajax.googleapis.com/ajax/libs/angularjs/1.4.7/angular-route.js>

