**Visual Studio Code with Node.JS**

**USING NODE JS**

--github.com/sayar/NodeMVA

https://www.youtube.com/watch?v=R9GYRBDqxI8&list=PLsrZV8shpwjNuvhvotOuzRCQGcqscCekJ

Popular NPM Modules

underscore

async

request

lodash

commander

express

optimist

coffe-script

NPM - Node Package Manager

- Bundled and installed automatically with the environment

Node Package Manager (npm), it provides a powerful environment to create platform-independent applications.

Frequent usage

- npm install - save package name

- npm update

How does it working

- Instal dependencies in the local node module folder

- In global mode , make a node module accessable to all

- can install from folder , tarbal and web etc

- can specify dev and optional depedencies

- NPM can also install the depedencies for a project by reading the package.json

Async Modules - utility module which provides straight forward , powerfull function for working with asynchronouse javascript

async.map

async.filter

async.parallel

async.series

Request Modules - Designated to be the simplest way possible to make http calls.

It supports HTTPS, streaming and follows redirect by default

Using Node.js with VS Code: 02 Introduction to Express

--github.com/sayar/NodeMVA

https://www.youtube.com/watch?v=GGrtjNbpEDo&index=2&list=PLsrZV8shpwjNuvhvotOuzRCQGcqscCekJ

What is Express

Installing and using express

DEMO : Creating a simple REST API

Templating

What is Express?

<https://www.youtube.com/watch?v=GGrtjNbpEDo&list=PLsrZV8shpwjNuvhvotOuzRCQGcqscCekJ&index=2>

To install all packages

Npm install express-generator –g

Express 🡨 not empty yes

Npm install

Copy the bin/www to the directory application

Then

Check the www port settings

<http://localhost:3001/>

- express is a minimal, open source and flexible node.js web.app frameworks designed to make developing websites, web apss and API's much easier

- Support multiple templating engines to simplify generating HTML

Installing Express

- npm install express

- npm install jade

02 - Creating a Simple REST API

Explanation of Routes

- A router maps HTTP request to a callback

- HTTP request can be sent as GET/POST/PUT/DELETE etc

- URLs describe the location of targeted

- Nodes helps you map a HTTP GET request likeL:

- http://localhost:8888/index

- To a request handler (callback)

app.get('/index', function(req,res){});

DEMO

**Using Express for Multiple Pages**

D:\Litopascua\GitHub\NodeJSLearn1\NodeJSLearn1\11\_ExpressMultipleRoutes\routes

Go to NodeJSLearn1 main directory

Npm install express-generator –g

Express

Do

Npm intall

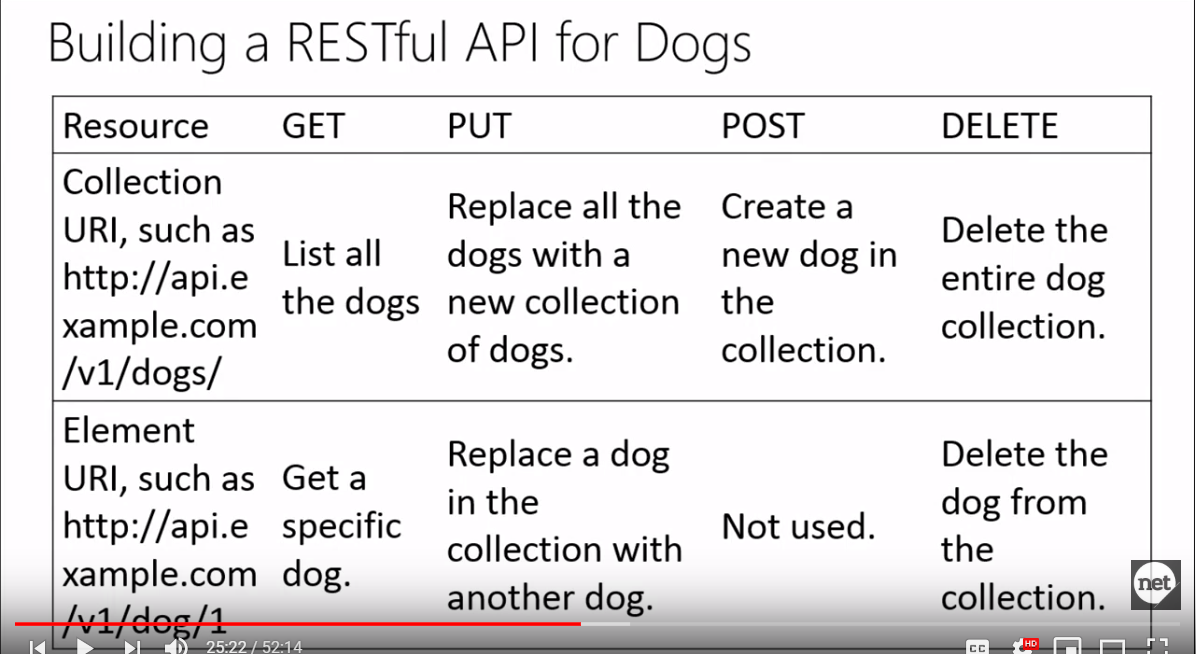
Npm start

Goto the

Localhost:3000

<https://www.youtube.com/watch?v=GGrtjNbpEDo&list=PLsrZV8shpwjNuvhvotOuzRCQGcqscCekJ&index=2>

BUILDING RESTful API for Dogs



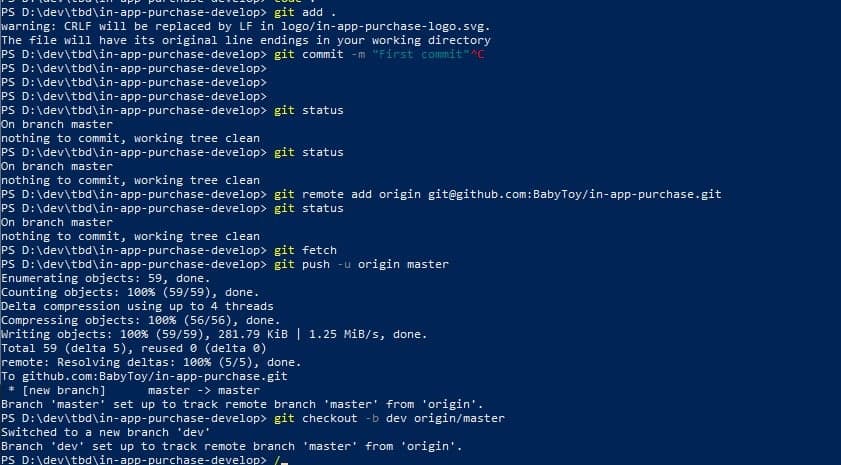
Express to Build a RESTful API

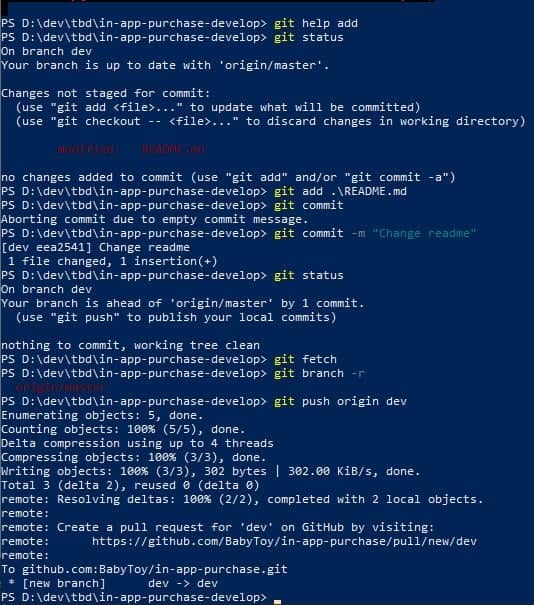
Resources

* Express Framework <http://expressjs.com>
* Intro to express
* Jade template <http://jade-lang.com/tutorials>
* Javascript and Jade templating

<http://www.slideshare.net/wearefractal/jade-javascript-templating>

* GitHub





Express to Build a RESTful API

Resources

<https://www.youtube.com/watch?v=GGrtjNbpEDo&list=PLsrZV8shpwjNuvhvotOuzRCQGcqscCekJ&index=2>

* Express Framework <http://expressjs.com>
* Intro to express

<https://code.tutsplus.com/tutorials/introduction-to-express--net-33367>

* Jade template <http://jade-lang.com/tutorials>
* Javascript and Jade templating

<http://www.slideshare.net/wearefractal/jade-javascript-templating>

use postman for Chrome

<https://www.youtube.com/watch?v=jAzmn3c3hac&list=PLsrZV8shpwjNuvhvotOuzRCQGcqscCekJ&index=3>

Using Node.js with VS Code: 03 Express and Databases

<https://www.youtube.com/watch?v=jAzmn3c3hac&list=PLsrZV8shpwjNuvhvotOuzRCQGcqscCekJ&index=3>

* MongoDB
* Data Driven Web Sites

Module Overview

* Discuss Data Driven Websites
* Discuss MongoDB
* Show you how to save to MongoDB using Mongoose

<https://www.youtube.com/watch?v=jAzmn3c3hac&list=PLsrZV8shpwjNuvhvotOuzRCQGcqscCekJ&index=3>

**MEAN** Stack – Mongo, Express, Angular and Node

About NoSQL databases

* **Not only SQL**
* Different types: Document based, graph databases etc.
* MongoDB, Couchbase, Hbase, Cassandra
* Object Oriented APIs
* Good for Large amounts of Data, can be scaled.

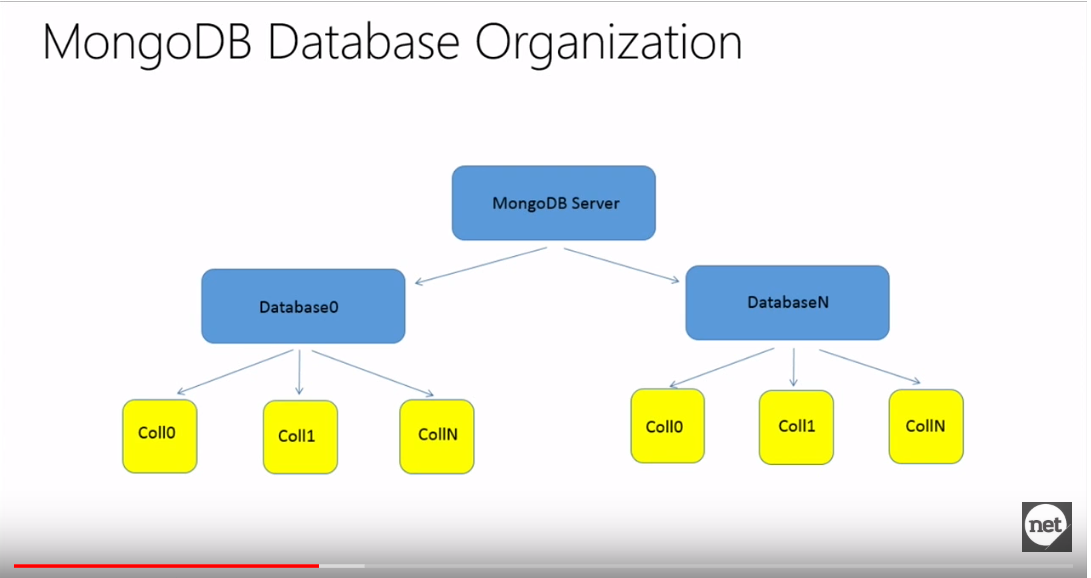
**Using MongoDB**

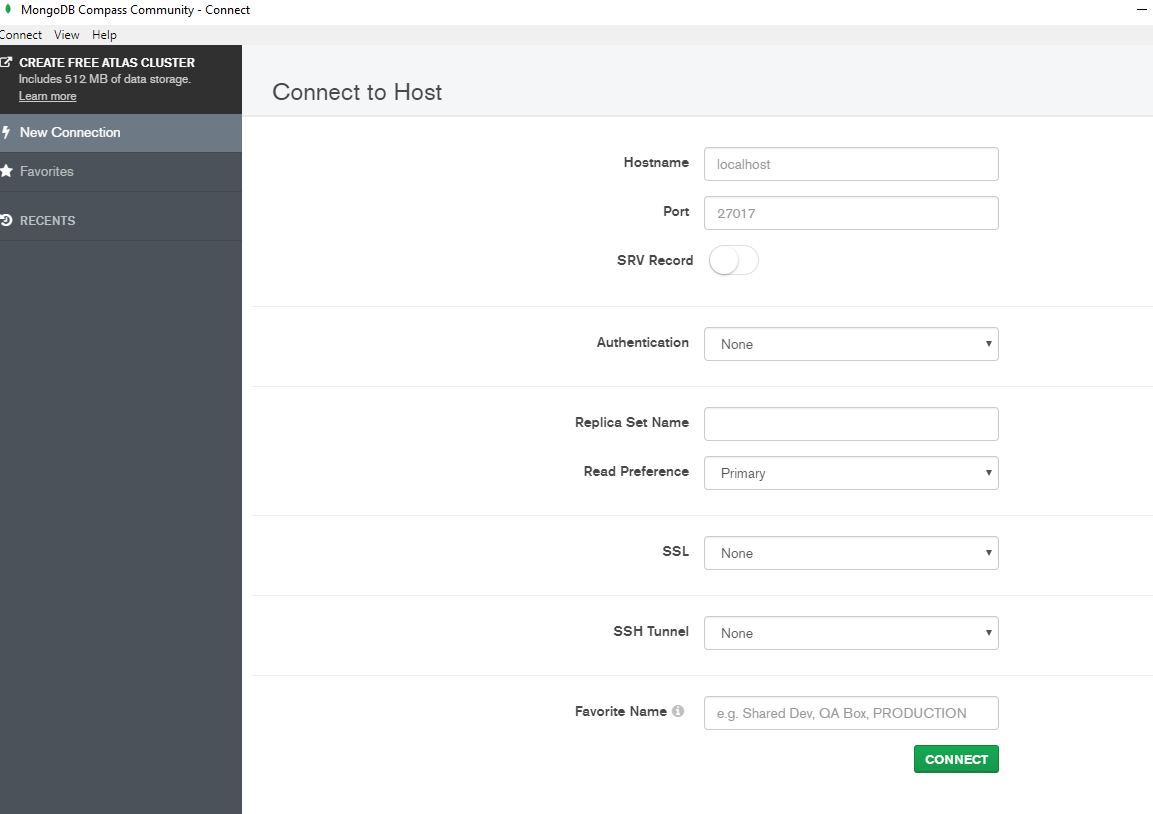
* Need to create mongoDB
* 3rd party source like MongoHQ or MongoLab
* Use the MongoLab add on in Azure
* Can customize and manually use via VM
* Need the MongoDB URI for connection

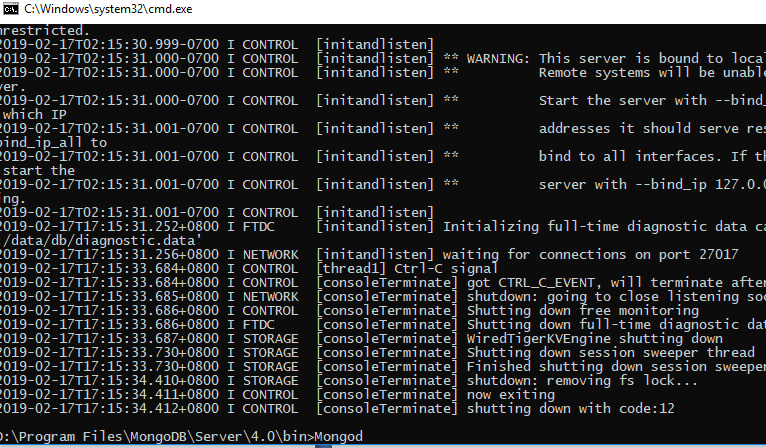
Running MongoDB on Local Machine

MongoDB Binaries

* The following binaries are installed in your machine
* Mongod – The database process
* Mongo – The Mongo CLI – used to connect
* Mongoimport – A data import utility







Loading Test Data

* Using mongoimport
* Load bulk data from CSV or JSON files
* Fastest way t o load data into mongodb without having to write code

Github source

<https://github.com/sedouard/mongodb-mva>

<https://github.com/sedouard/mongodb-mva/tree/master/module2_getting_started>

## Loading up test data into mongodb

Within the data directory of this repository there is a **bank\_data.json.zip** file; you should decompress it. We will use this file to load up the bank data collection into mongodb. From the mongodb installation **bin** in a new console window, execute the command:

mongoimport <path to bank\_data.json> --jsonArray --collection bank\_data

our directory

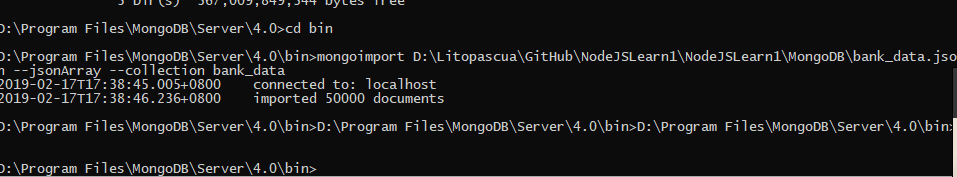
D:\Litopascua\GitHub\NodeJSLearn1\NodeJSLearn1\MongoDB

Bin path

D:\Program Files\MongoDB\Server\4.0\bin>

* Import the json file bank\_data.json

mongoimport D:\Litopascua\GitHub\NodeJSLearn1\NodeJSLearn1\MongoDB\bank\_data.json --jsonArray --collection bank\_data



**Collections**

* Container for a group of Documents
* Database contain many of these
* Akin to a ‘Table’ in a SQL Database

**Queries**

* User a Query Object to Fetch Requested Data
* Resembles objects already in the database
* Can also be used to Update or Delete data

**Projections**

* Used to filter the date you want
* Similar to what SELECT does in SQL query
* Take similar form of Query objects
* Start Mongod database listening to port 12017
* Node app.js

D:\Litopascua\GitHub\NodeJSLearn1\NodeJSLearn1\13\_MongoDB

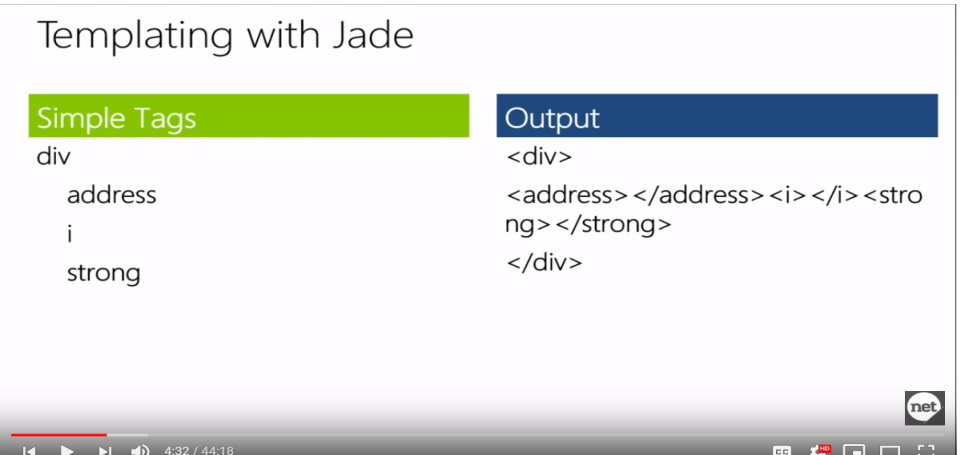
Using Node.js with VS Code: 04 Building a Front End for Your Express Web

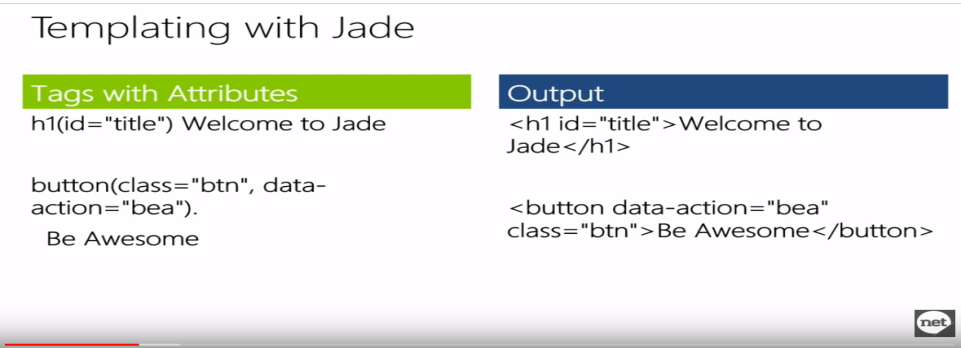
<https://www.youtube.com/watch?v=ZnTVyzGcVNA&list=PLsrZV8shpwjNuvhvotOuzRCQGcqscCekJ&index=4>

* **Introduction to Jade**
* Jade Templates
* Implementing Bootstrap
* Express and the template language **Pug** (formerly known as JADE).

**Templating with Jade**

* Jade is a templating language to simply writing HTML
* Jade syntax and keywords map directly to HTML
* Jade add the ability to seperate and extend HTML
* Help prevent code repeat
* Ensure clean HTML is generated
* Allows you to insert values into HTML through templates





The extend keywords allows a template to extend a layout or parent template. It can then override certain pre-defined blocks of content.

The block keyword allows you to establish a block or replace the content of pre-defined blocks.

Templating with Jade

The **extends** keywords allow a template to extend a layout or parent template. It can the override certain pre-defined blocks of content.

The **block** keyword allows you to establish a block or replace the content of pre-defined blocks.

**Using Jade Templates**

**DEMO**

**D:\Litopascua\GitHub\NodeJSLearn1\NodeJSLearn1**

[**http://jade-lang.com/**](http://jade-lang.com/)

[**http://localhost:3001/**](http://localhost:3001/)

To install all packages

Npm install express-generator –g

Express 🡨 not empty yes

Npm install

Copy the bin/www to the directory application

Then

Check the www port settings

<http://localhost:3001/>

**Bootstrap?**

* Framework for developing responsive site including presentation and behavior (HTML/CSS/JS)
* Number 1 project in GitHub
* <http://getbootstrap.com/>
* Accounts for common functionality and layout needs
* Customize it to the project needs:

<http://getbootstrap.com/customize/>

Bootstrap Resources

Microsoft Virtual Academy

* <http://aka.ms/bldUI-bootstrp>

edX

* <https://www.edx.org/course/introduction-bootstrap-tutorial-microsoft-dev203x-0>

Using Node.js with VS Code: 05 Debugging and Deploying Node.js

<https://www.youtube.com/watch?v=CjlBlVMGbA8&list=PLsrZV8shpwjNuvhvotOuzRCQGcqscCekJ&index=5>

Module Overview

* Azure Website Overview
* Introduction to the Azure Dashboard
* Deploying to Azure with the command line
* Deploying to Azure with Github
* Advanced Debugging

Azure Web Apps

* Easy to deploy a variety of different web site : node, python, php, asp.net,etc
* Can install some software from the gallery like WordPress or preconfigured stacks( MEAN Stack)
* Has a few limitations such as cannot configure ports, compile, native modules for Node.

Azure Web Apps

* Can configure custom domains
* Can Add FTP Users
* Can Run Multiple web sites
* Ideal for a production / staging environment

Introduction to the Azure Dashboard

This cloud computing service is a big part of Microsoft’s business, and it competes with similar services from Amazon and Google.

05 Advanced Debugging

<https://www.youtube.com/watch?v=CjlBlVMGbA8&list=PLsrZV8shpwjNuvhvotOuzRCQGcqscCekJ&index=5>

easy debugging using vscode

# 06 Using Node.js with VS Code: 06 Extending Node.js with Azure Web Jobs

<https://www.youtube.com/watch?v=mrhRsw1SFSk&list=PLsrZV8shpwjNuvhvotOuzRCQGcqscCekJ&index=6>

Module Overview

* Azure WebJobs Overview
* Introduction to Creating WebJobs
* Creating Simple Webjobs
* Using Azure Storage in a WebJob
* Debugging a Webjob

Introduction to Creating Webjobs for Node

* Look for a file with “run” (run.js). If not found looks for any file with supported extensions.
* Have a many files as your want, you zip them up
* Can have multiple webjobs
* Stored alongside your site at ./App\_Data/jobs/

=== **Programming Web Applications with Node, Express and Pug,  ===**

<https://scb.percipio.com/books/f7d11800-fb80-11e6-8e3a-0242c0a80a04>

It's necessary to have a well-known language to create web applications instead of inventing just another language all the time. Think of the history of Perl, PHP, Java, Ruby, C in all it derivates, and many more. The final answer might be **JavaScript.**

The **foundation of JavaScript** on the server is the **V8 engine from Chrome browser** that has been extracted and made available as an executable. And it is available on all platforms, finally.

For developers with a strong background in traditional object-oriented languages such as C# or Java, it might sound odd that JavaScript has such a tremendous impact and success. It's a weird mixture of a very simple language and a very rich and quickly expanding ecosystem.

### Basis Libraries of the Server

In this book series a form of the MEAN stack is presented. MEAN stands for:

* MongoDB/MySQL
* Express
* AngularJS
* Node

That is striking, but only the half truth. The choice of the database is often not primary and most components are often not sufficient in order to illustrate the entire Web stack. It should be considered for the server, that:

* As server page, Routing Framework and **Express** middleware is used. It supplies the Routing functions and is an efficient application framework.
* As Template library, **Pug** is used, which takes over the production of the HTML forms instead of Razor, as far as this takes place on the server.

### Client Page Libraries

Thus we can deliver web pages and make services available. The client support remains:

* **AngularJS** as the comprehensive framework for the structuring of the pages
* **Bootstrap** as design and style framework
* **jQuery** as implicitly library used by Bootstrap for the access to the Document Object Model (DOM)

All of this would also be used in the ASP.NET world. Here .NET offers no direct entrance, because the client can be served only over JavaScript.

### Web Apps

Applications are called Web Apps if they exist directly in the browser and communicate only with the server in order to reload data dynamically. The server thereby first delivers the app and then supports it by services, for example to the access of a database. The server places thereby a so called API (Application Programming Interface) as available. Usually this is based on JSON.

### Web sites

Most web sites are rather classically programmed. That means the detectability of contents through search engines, extremely short load times, and simple structure. The server produces finished HTML and all dynamic elements by manipulating the HTML with the help of small scripts. Forms are used for interaction and the indicator functions by the server steered. Web sites are then supported by JavaScript so that they appear interactive, which is necessary in order to appear modern and functional.

However, this approach is problematic for several reasons. They must hold two code environments separately from each other: on **the one hand for the browser**, **on the other hand for the server**. Both worlds are closely connected. Changes on one page can release errors on the other page. This entwinement is critical and hardly permanently controllable.

### Stateless HTML

If Web Apps are not an option (complex, slow, not a search engine suite) and also not web sites (maintenance-unfriendly, faulted), then it is time to think about a new strategy. **This is where Node comes** in, because the separation of the code environments is by far less drastic, if the same programming language is used. Additionally, a certain programming style should be used. This is so-called stateless HTML.

Stateless HTML is a piece of HTML that is always identical to the condition of the web site and independent. Whether the user is registered or not, whether it is morning or afternoon, it is all the same. No matter which geographical place was used, the HTML of the page is always alike. Thus a significant part of maintenance cost is lost. Parts of the page, which are dependent on the user or action, do not become part of the HTML. They are procured like a Web App by services and provided dynamically. Thus, simple loading from HTML pages is in Node, as in the examples shown.

Imagine a page with contents, which readers can discuss. The contents part is for all users directly. Also, each search engine sees the same contents. This part is static and condition-independent. That does not mean that the articles must lie statically on the hard disk. They can be assembled on the server from a database. It is part of the panel and completely dynamic against it. Each user sees his own contributions differently and has perhaps personalized the representations. This part is provided and delivered differently.

The approach does not only simplify programming. It also increases the performance clearly. The less dynamic portion is easier to process on the server and on the client. A cache can be used comprehensively and be further relieved from the server. Also, in the event of an error delivering of static pages, it is more robust and more reliable. The omission of the dynamic functions is annoying, but the page remains complete and searchable. However, an improvement of the user experience is crucial.

<https://scb.percipio.com/books/f7d11800-fb80-11e6-8e3a-0242c0a80a04>