Building a Web Application

with MEAN stack and others

Babun (Windows) http://babun.github.io Terminal (Mac)

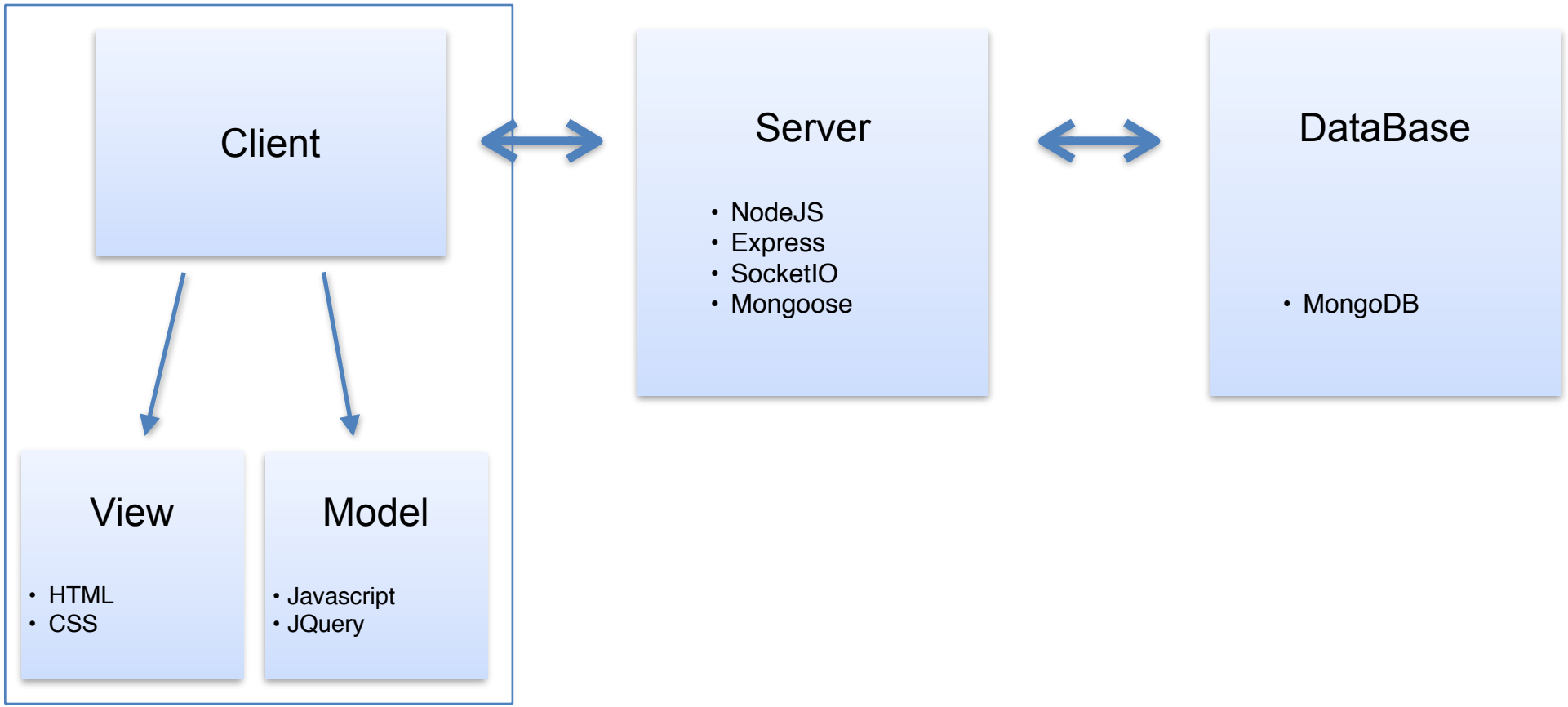
GitHub https://github.com/joanchen306/DIntern-2015

Sublime Text http://www.sublimetext.com

HomeBrew (Mac) http://brew.sh

Web Application - Architecture

網頁的架構：



Web Application - Client

Model v.s. View:

View (User Interface)

```
<body>
  <input type='text' id='textInput' style='width:500px'/>
  <input type='button' value='click' id='clickBtn' />
</body>
```

Model (Logic, Functions)

```
<head>
  <script src='js/jquery-2.1.4.min.js'></script>
  <script>
    $(function(){
        $('#clickBtn').click(function(){
          console.log($('#textInput').val());
        })
    })
  </script>
</head>
```

Web Application-View

HTML, CSS:

- W3Schools: http://www.w3schools.com/html/default.asp

w3schools.com

- Jade (Express)

JADE
NODE TEMPLATE ENGINE

```
doctype html
html(lang="en")
  head
    title= pageTitle
    script(type='text/javascript').
      if (foo) {
        bar(1 + 5)
      }
  body
    h1 Jade - node template engine
    #container.col
      if youAreUsingJade
        p You are amazing
      else
        p Get on it!
      p.
        Jade is a terse and simple
        templating language with a
        strong focus on performance
        and powerful features.
```

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Jade</title>
    <script type="text/javascript">
      if (foo) {
        bar(1 + 5)
      }
    </script>
  </head>
  <body>
    <h1>Jade - node template engine</h1>
    <div id="container" class="col">
      <p>You are amazing</p>
      <p>
        Jade is a terse and simple
        templating language with a
        strong focus on performance
        and powerful features.
      </p>
    </div>
  </body>
</html>
```

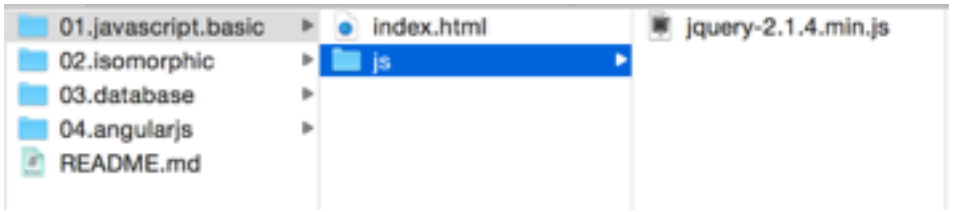
Web Application - Model

JavaScript

Controls the **behavior** of the Web Page

```
<!DOCTYPE html>
<html>
  <head>
    <script>
    function myFunction() {
        document.getElementById("demo").innerHTML = "Paragraph changed.";
    }
    </script>
  </head>
  <body>
    <p id="demo">A Paragraph.</p>
    <button type="button" onclick="myFunction()">Try it</button>
  </body>
</html>
```

JQuery https://jquery.com W3School-jQuery




```
<head>
  <script src='js/jquery-2.1.4.min.js'></script>
  <script>
    $(function(){
```

Node JS and Express

- install NodeJS
- install npm

```
$ npm init
```

  - package.json : manage your package.
- install Express

```
$ npm install express
```

```
$ npm install express-generator@4
```

Web Application - Server

Node JS

- Create our server

/bin/WWW

```
var app = require('../app');
var http = require('http');
var server = http.createServer(app);

server.listen(port);
server.on('error', onError);
server.on('listening', onListening);
```

Web Application - Server

- Routes
  - display
  - data communicate
- Views
  - layout.jade
  - index.jade
  - error.jade

Web Application - Server

Express

- Set up path

app.js

```
var express = require('express');
var routes = require('./routes/index');

var app = express();

// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'jade');
app.use('/', routes);
```

Express

- Set up our routes

routes/index.js

```
var express = require('express');
var router = express.Router();

/* GET home page. */
router.get('/', function(req, res, next) {
  res.render('index', { title: 'Express' });
});

module.exports = router;
```

Web Application - Sockets

Socket IO - Real-Time Engine http://socket.io

```
$ npm install --save socket.io
```

./bin/www

```
#!/usr/bin/env node
var io = require('socket.io');
```

```
/**
 * Create HTTP server.
 */

var server = http.createServer(app);
var pos = {top:0, left:0};
console.log(pos);

io = io.listen(server);
io.on('connection', function(socket){
  console.log('User connected. Socket id %s', socket.id);
  io.emit('update', pos);

  socket.on('disconnect', function(){
    console.log('User disconnected. Socket id %s', socket.id);
  });

  socket.on('update', function(_pos){
    pos = _pos;
    io.emit('update', pos);
  });
});
```

Web Application - Sockets

layout.jade

```
head
  title=title
  script(src='/socket.io/socket.io.js')
```

index.jade || *.jade

```
block content
  script.
    $(function(){
      var socket = io();
```

```
socket.on('chat message', function(msg){
```

```
$('#msgBtn').click(function(){
  if ($('#msgInput').val() != '') {
    var msg = $('#msgInput').val() + '\n';
    var allMsg = $('#msgTextarea').val() + '#{name}' + ' : ' + msg;
    socket.emit('chat message', allMsg);
    $('#msgInput').val('');
    $.post(window.location.href, { name: '#{name}', chat: msg});
  }
});
```

MongoDB and Mongoose

- install MongoDB

```
$ npm install mongodb
```

- install Mongoose

```
$ npm install mongoose
```

Web Application - Database

MongoDB and Mongoose

- Database
- Folders
  - models

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

var TodoSchema = new mongoose.Schema({
  name: String,
  completed: Boolean,
  note: String,
  updated_at: {type:Date, default: Date.now}
});

module.exports = mongoose.model('Todo', TodoSchema);
```

MongoDB and Mongoose

- Database communication
- RESTful

GET	POST	PUT	DELETE
Get data	Create data	Update data	Remove data

MongoDB and Mongoose

- Update our Routes/
- RESTful support

```
var express = require('express');
var router = express.Router();
var mongoose = require('mongoose');
var Todo = require('../models/todo.js');

router.get('/', function(req, res, next) {
  Todo.find(function(err, todos){
    if(err) return next(err);
    res.json(todos);
  })
});

router.post('/', function(req, res, next) {
  Todo.create(req.body, function(err, post){
    if(err) return next(err);
    res.json(post);
  })
})
```

MongoDB and Mongoose

- RESTful support

```
router.get('/:id', function(req, res, next){
  Todo.findById(req.params.id, function(err, post){
    if(err) return next(err);
    res.json(post);
  })
})

router.put('/:id', function(req, res, next){
  Todo.findByIdAndUpdate(req.params.id, req.body, function(err, post){
    if(err) return next(err);
    res.json(post);
  })
})

router.delete('/:id', function(req, res, next){
  Todo.findByIdAndRemove(req.params.id, req.body, function(err, post){
    if(err) return next(err);
    res.json(post);
  })
})
```

Smarter. Greener. Together.

To learn more about Delta, please visit www.deltaww.com.