ADALTAS

WEBSOCKETS

ASYNCHRONOUS SERVER TECHNOLOGIES

César Berezowski

Big Data Consultant @ Adaltas

cesar@adaltas.com

ADALTAS

UP UNTIL NOW

- NodeJS: simple HTTP server with your modules
- NPM to manage our application and modules
- ExpressJS:
  - Handle routing
  - Manage user auth and sessions with middlewares
  - Expose static content to display in a browser
- Transpilers to ease writing and reading code
- Level DB to store the data
- ...

And also a set of best-practices and other tools to enhance the developer's experience

ADALTAS

FINAL PROJECT

- Using the code from class
- Simple dashboard app :
  - User login
  - A user can insert metrics
  - A user can retrieve his metrics in a graph
  - A user can only access his own metrics

ADALTAS

QUESTIONS ?

Now is the time !

ADALTAS

WEBSOCKETS

ADALTAS

CLIENT-SERVER COMMUNICATION

- Server holding the data
- Client getting the data
- Client needs to call to check for update
- Client needs to send its updates
- Regular updates => regular calls to the server
- Only simulated realtime with HTTP Polling

ADALTAS

WHAT ARE WEBSOCKETS ?

- Client/server communication technology
- TCP socket connection between client & server
- Bi-directional => full duplex
- Low latency connection

ADALTAS

HOW DO WE USE THEM ?

- Socket.IO !
- JS library
- Two part library:
  - client in browser
  - server side in Node.JS
- Event-driven, like Node.JS
- Used in Microsoft Office, Yammer, Zendesk, ...

QUESTIONS ?

SOCKET.IO SETUP

ADALTAS

BASIC EXPRESS SERVER

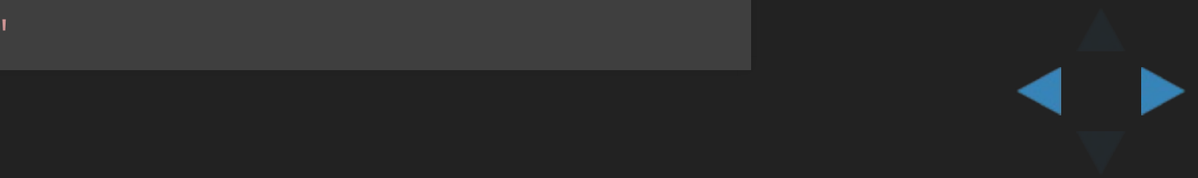
- npm init
- npm i --save express coffee-script jade jstransformer-coffee-script jstransformer-stylus nodemon

```
# src/app.coffee
app = require('express')()

app.set 'port', 1337
app.set 'views', "#{__dirname}/../views"
app.set 'view engine', 'jade'

app.get '/', (req, res) ->
  res.render 'index'

app.listen app.get('port'), ->
  console.log "server listening on #{app.get 'port'}"
```



ADALTAS

SERVER WITH SOCKET.IO

- npm i --save socket.io
- Wrap express with node-http
- Instantiate socket.io with the server as parameter

```
# src/app.coffe
eapp = require('express')()
server = require('http').Server(app)
io = require('socket.io')(server)
```

ADALTAS

SERVER WITH SOCKET.IO

- Open socket.io connection
- Prepare events

```
# src/app.coffe
express = require 'express'
app = express()
server = require('http').Server(app)
io = require('socket.io')(server)

# Rest of server code

io.on 'connection', (socket) ->
  socket.emit 'news',
    hello: 'world'
  socket.on 'client event', (data) ->
    console.log data

server.listen app.get('port'), ->
  console.log "server listening on #{app.get 'port'}"
```

ADALTAS

FRONT

Require /socket.io/socket.io.js

```
# views/index.jade
block head
  script(type="text/javascript" src="/socket.io/socket.io.js" charset="utf-8")
```

Connect to the socket

```
# views/index.jade
block content
  p Hello
  script
    :coffee-script
      socket = io.connect 'http://localhost:1337'
      socket.on 'news', (data) ->
        console.log data
        socket.emit 'client event', my: 'data'
```

CHECK OUR INSTALLATION

- ./node_modules/.bin/nodemon src/app.coffee
- Load http://localhost:1337 in your browser
- Check Node.JS' and the browser's console

ADALTAS

LET'S MAKE A SIMPLE CHAT

https://github.com/cesarBere/ece-nodejs-chat

ADALTAS

SOCKET.IO IN OUR CONTEXT

We'll use it for logging

ADALTAS

SERVER SIDE

Setup the Socket.IO server

```
app = express()
server = require('http').Server(app)
io = require('socket.io')(server)

sockets = []

io.on 'connection', (socket) ->
  sockets.push socket

# Session loading
```

ADALTAS

SERVER SIDE

Prepare a logging middleware with IO

```
# Logging middleware
app.use (req, res, next) ->
  for socket, i in sockets
    socket.emit 'logs',
      username:
        if req.session.user == undefined then 'anonymous'
        else req.session.user.username
      url: req.url
  next()
```

SERVER SIDE

Set logging route and run the server

```
app.get '/logging', (req, res) ->
  res.render 'logging'

server.listen '1337', ->
  console.log "listening on port 1337"
```

ADALTAS

FRONT SIDE

Prepare a page to display the logs

```
doctype html
 html(lang="en")
  head
   title AST - Logs
   link(rel='stylesheet', href='/css/bootstrap.css')
   script(type="text/javascript" src='/js/jquery-2.1.4.min.js')
   script(type="text/javascript" src='/js/bootstrap.min.js')
   script(type="text/javascript" src='/socket.io/socket.io.js')
  body
   .container
    h1 Logs
    #content
```

ADALTAS

FRONT SIDE

Setup the JS to handle Socket.IO

```
script
  :coffee-script
    socket = io.connect 'http://localhost:1337'
    socket.on 'logs', (data) ->
      $('#content').append "#{data.username} loaded #{data.url}"
    ...
```