

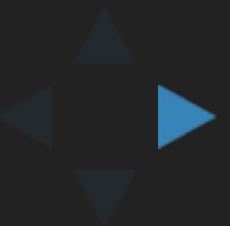
WEBSOCKETS

ASYNCHRONOUS SERVER TECHNOLOGIES

César Berezowski

Big Data Consultant @ Adaltas

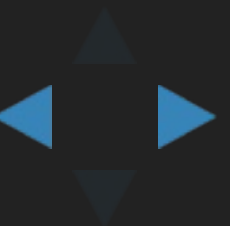
cesar@adaltas.com



UP UNTIL NOW

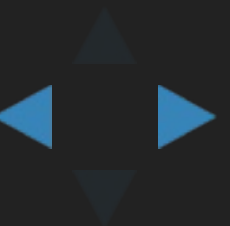
- NodeJS: simple HTTP server with your modules
- NPM to manage our application and modules
- ExpressJS:
 - Handle routing
 - Manage user auth and sessions with middlewares
 - Expose static content to display in a browser
- Transpilers to ease writing and reading code
- Level DB to store the data
- ...

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



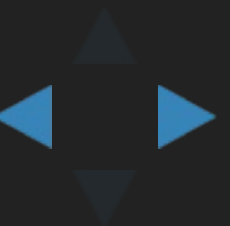
FINAL PROJECT

- 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



QUESTIONS ?

Now is the time !

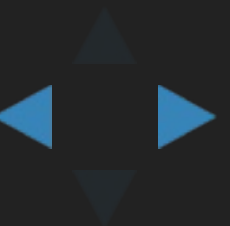


WEBSOCKETS



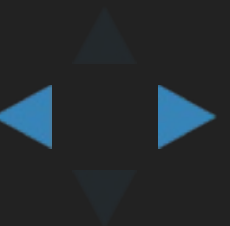
CLIENT-SERVER COMMUNICATION

- 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



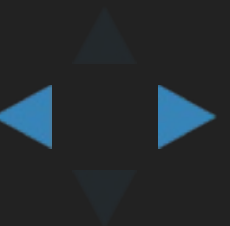
WHAT ARE WEBSOCKETS ?

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

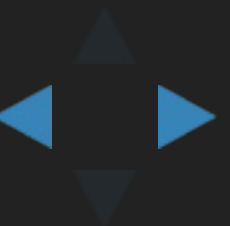


HOW DO WE USE THEM ?

- 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



BASIC EXPRESS SERVER

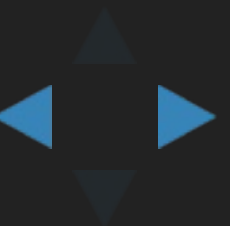
- npm init
- npm i --save express coffee-script jade jstransformer-coffee-script jstransformer-stylus nodemon

```
# 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'

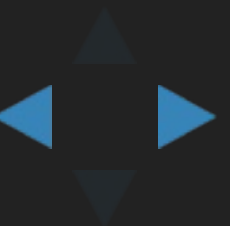
app.listen app.get('port'), ->
  console.log "server listening on #{app.get 'port'}"
```



SERVER WITH SOCKET.IO

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

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



SERVER WITH SOCKET.IO

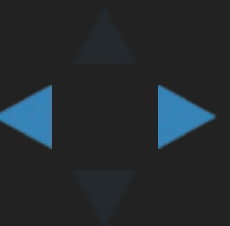
- Open socket.io connection
- Prepare events

```
# 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'}"
```



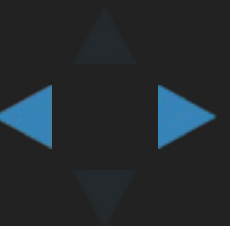
FRONT

Require /socket.io/socket.io.js

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

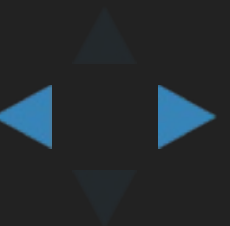
Connect to the socket

```
# 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



LET'S MAKE A SIMPLE CHAT

