Node.JS > INTRODUCTION

TUTORIAL

In this tutorial we will build an Multi-Touch Interface which can be used to control Ableton Live.

We will

- get an overview about the possibilities with node.js
- learn the basics of Javascript
- learn how to implement a node.js script in Max for Live
- create a UI with p5.js
- implement a Webserver via node.js
- use websocket communication protocol to exchange data from client to server

Possibilities of Node.JS

NODE.JS -> Max for Live Showreel -> https://www.youtube.com/watch?v=gEtONnWOGyo

Cloud Azure & AWS

Al & Machine Learning • Analytics • Blockchain • Compute Containers • Databases • Developer Tools • DevOps Identity • Integration • Internet of things Management and Governance • Media • Migration • Mixed Reality Mobile • Networking • Security • Web • Virtual Desktops

APIs

Facebook • Twitter • Giphy • Freesound • Archive.org
Twilio • Nexmo • WebRTC • IFTTT • Netflix • Dominos
Cats • Film • Science • NASA • Transit • City Data
Wikipedia • Soundcloud • IBM • Paypal • Stripe
Weather • Sports • Dictionary • TensorFlow • OpenCV
FFmpeg • FFprobe • ImageMagick • Imgur • Cylon.js
Periscope • Twitch • YouTube • Apple • HVAC
Environmental • MailChimp • Authy • loT products • LIFX

Artificial Intelligence

Machine Learning • Deep Learning • Supervised/Unsupervised Natural Language Processing • Content Extraction • Classification Machien Translation • Question Answering • Text Generation Expert Systems • Vision • Image Recognition • Machine Vision Speech • Speech to Text • Text to Speech Planning • Robotics

Web Frameworks

Express • React • Angular • Vue Meteor • Ember • Backbone Aurelia • Polymer

CSS Frameworks

Bootstrap • Foundation • Bulma Skeleton • Pure • Groundwork Cardinal • Meuller • Materialize • Semantic

Compositions

Interactive Environments • Global instruments
No-border composition • Create & upload feedback loops
Elapsed composition • Additive data compositions

IoT

WiFi IoT • Cellular IoT • Global connectivity
CoAP • UDP/TCP • MOTT



MySQL • MongoDB • PostgreSQL • SQLite • MariaDB Oracle • Couchbase • Cassandra • LocustDB • Redis Amazon Web Services • Azure • Digital Ocean

Databases

Robotics

Teleoperations • Remote data collection autonomous interaction Telerobotics • Remote Bioinformatics

Web Browser/Mobile

WebSockets • HTTP + JSON • Event Triggering

Topics

IoT • Authentication • Data Backups • Serverless
Fog Computing • Ngrok • WebSockets • WebRTC • HTTP Requests •
CoAP • Telecom • Web Scraping • Exploits • Static-site • Website w/Routing
Web Interactivity • Video Streaming • Audio Streaming • Chatbots
Text to Speech • Speech to Text • Robotics • Data Encryption
Audio Encryption • Cron jobs (event scheduling)
Push Notifications • Proxies • Server Filesystem • Scalability • Analytics
Data Mining • Data research • Remote data acquisition • Blockchain
Security • Backend Processing • Bioinformatics • Hardware interfacing

Services

Microsoft Flow • IFTTT • Zapier

Gettings Started:

- https://cycling74.com/articles/node-for-max-intro---let's-get-started
- Node for Max Documentation -> https://docs.cycling74.com/nodeformax/api/index.html

About Node:

- https://nodejs.org/en/
- https://www.npmjs.com

EXAMPLES:

- face API connection with Max-> https://www.youtube.com/watch?v=Ud XvmPlavU
- How to track human skeleton
 - https://mediapipe.dev
 Skeletontracking example p5.js -> https://editor.p5js.org/lingdong/sketches/ef6FB-uNq
 Handtracking example p5.js -> https://editor.p5js.org/lingdong/sketches/1viPqbRMv

Node for Max Resources

- Gettings Started https://cycling74.com/articles/node-for-max-intro----let's-get-started
- Node Max Showreel https://www.youtube.com/watch?v=qEtONnWOGyo
- Node for Max: An Introduction https://www.youtube.com/watch?v=qSZH6fjOcXE
- Getting Started https://www.youtube.com/watch?v=QulcEHJSwz8
- Node For Max Core Examples https://github.com/Cycling74/n4m-core-examples/
- Node For Max Examples https://github.com/cycling74/n4m-examples

Recommended EDITOR'S

- Visual Studio Code https://code.visualstudio.com
- Sublime https://www.sublimetext.com
- atom.io https://atom.io

How to add a node.js script in Max for Live

- 1. new Object -> node.script hello.js
- 2. Connect message Box -> script start
- 3. Create new js file hello.js at harddisk (same directory as M4L device)
- 4. Add following lines into the hello.js file

```
// include max-api -> is the connection to Max const maxAPI = require('max-api'); 
// print Message -> "Hello from Node.js" to Max-Console maxAPI.post("Hello from Node.js");
```

- 5. Trigger message Box "script start"
- 6. Check if you get a message at Max Console

Node.js - Basics

Add a handler:

```
// a handler for receiving a message with a value e.g. "slider 60"
maxAPI.addHandler('slider', (value) => {
    //print value
    maxAPI.post(value);
});

// a handler for receiving a message without a value
maxAPI.addHandler('generateRandomNumber', () => {
    let rand_value = Math.random()
    maxAPI.post(rand_value);
});
```

Create a connection to the outside:

```
// send a message "random_value 0.234" to the outlet maxAPI.outlet('random_value', 0.234);

// print "Hello" to Max Console maxAPI.post("Hello");
```

Create a function:

```
function calc() {
    //do some calculations
}
```

Creating a variable:

let value = 10;

Debugging:

- use node.debug Object and connect it to the most right outlet of

EXERCISE: create a new node.js script which receives 2 values and multiply these values when one value will be received. Output the result as a message "result 6" if the input was 2 and 3

Javascript Cheatsheet http://www.developer-cheatsheets.com/es6

INTRO P5.JS

Projectpage https://p5js.org

Recommended tutorials:

- https://thecodingtrain.com/beginners/p5js/

P5.js Getting started:

- https://p5js.org/get-started/

Create UI:

BASIC -> https://p5js.org/examples/dom-slider.html MULTITOUCH .> https://github.com/L05/p5.touchgui

YOU CAN FIND THE CODE FOR THE TOUCH UI IN FOLDER

- p5.js ctrl interface > index.html (template)
- copy the file to your webserver (M4L or native) in the "puplic" folder.

WEBSERVER

Definition: https://en.wikipedia.org/wiki/Web_server

If you need a Webserver for experiments you can use python.

To start a webserver run the command below:

python3 -m http.server

That will open a webserver on port 8080. You can then open your browser at http://127.0.0.1:8080/ The webserver is also accessible over the network using your 192.168.-.- address. This is a default server that you can use to download files from the machine.

In class:

- Get access to p5 files from the Webserver
- get acces to files via smartphone

CREATING a NODE.JS WEBSERVER WITH EXPRESS

M4L Implementation Ready to GO:

- OPEN > folder AbletonLive_M4L_connection > M4L_Server > Node_JS_Server.amxd

DETAILS

Adapted from Daniel Shiffman: https://www.youtube.com/watch?v=2hhEOGXcCvg

- 1. Install Node.JS
 - Windows https://nodeis.org/en/download/package-manager/#windows
 - 2. MAC https://nodejs.org/en/download/package-manager/#macos
- 2. Check if node.js is installed
- OPEN your Terminal and type
 - node -v
- 3. Check if nom is installed and type
- npm -v
- 4. Create a folder "server"
- 5. Change directory to server
- 6. Create a file -> server.is
- 7. Type command -> npm init(setup a configuration file package.json)
- 8. npm install express —save (install express Webserver and add dependency in package.json)
- 9. Add following lines into server.js

```
//create webserver and listen at port 3000
var express = require('express');
var app = express();
// listen at port 3000
var server = app.listen(3000);
// make all files in directory public accessible (these files are static files, index.html. jsfiles, media)
app.use(express.static('public'));
// print message at Terminal
console.log("socket server is running");
```

- 10. Execute browser with url -> http://localhost:3000
- 11. Create directory "public" in directory "server"
- 12. Add index.html and p5.js files in public directory
- 13. Test with url if public content will be loaded
- 14. Install Websocket package socket.io
 - npm install socket.io -save
- 15. Add web socket code to server.js

```
//create websocket
var socket = require('socket.io');
//add server to the socket
var io = socket(server);
//deal with events - in these case "connection" and call a function when connection is working
io.sockets.on('connection', newConnection);
```

- 16. Add socket client library to index.html file
 - https://socket.io/docs/v4/client-api/
 - -<script src="/socket.io/socket.io.js"></script>
- 17. Create the connection from the client to the server and add following lines in sketch.js
 - var socket:
 - in setup function > socket = io.connect('http://localhost:3000');

```
18. Add following code that you can communicate with the server in sketch.js
       //create socket connection
       var socket;
       var websocket_msg = {
        name: ",
        value: null
       sendWebSocketMSG(websocket_msg);
       function sendWebSocketMSG(data) {
        socket.emit('msg', data);
Comment: value could also be a complex data set like
       mouse_data = {
         x: 0,
         y: 0
       mouse_data.x = 23;
       mouse_data.y = 60;
19. Add following code to your server (server.js) for receiving messages from the client
       function newConnection(socket) {
         console.log('new connection: ' + socket.id);
         socket.on('msg', getMsg);
         function getMsg(data) {
            if (DEBUG) {
              console.log(data);
            }
            if (data.name == 'mouse_data') {
            socket.broadcast.emit('msg', data);
       }
```

Receive values in M4L

OPEN > Folder websocket_client > websocket_client_interface.amxd

Install dependencies > trigger following message boxes script npm init script npm install socket.io-client --save

Toggle > START

Smartphone Connection

- · Your smartphone need to be in the same local network as the server
- Open the file index.html in folder "public" at the server and add the IP-Address of your local machine. Edit following line and change
 - socket = io.connect('http://localhost:3000'); to e.g.
 - socket = io.connect('http://192.168.3.1:3000');
 - Restart the server and open the page http://192.168.3.1:3000 in a browser which supports websockets

USAGE

EXAMPLE: EXHIBITION
COMPUTER IS RUNNING IN AN EXHIBITION
WEBSERVER IS RUNNING ON THAT COMPUTER
SCAN QR CODE > OPENS WEBPAGE > GET CONTROLLER INTERFACE

Further Ressources

HOW TO ADD A NODE PACKAGE VIA NPM

https://docs.cycling74.com/max8/vignettes/02 n4m usingnpm

In our case a web socket client: (We are using <u>socket.io</u>-client) <u>https://www.npmjs.com/package/socket.io-client</u>