Web

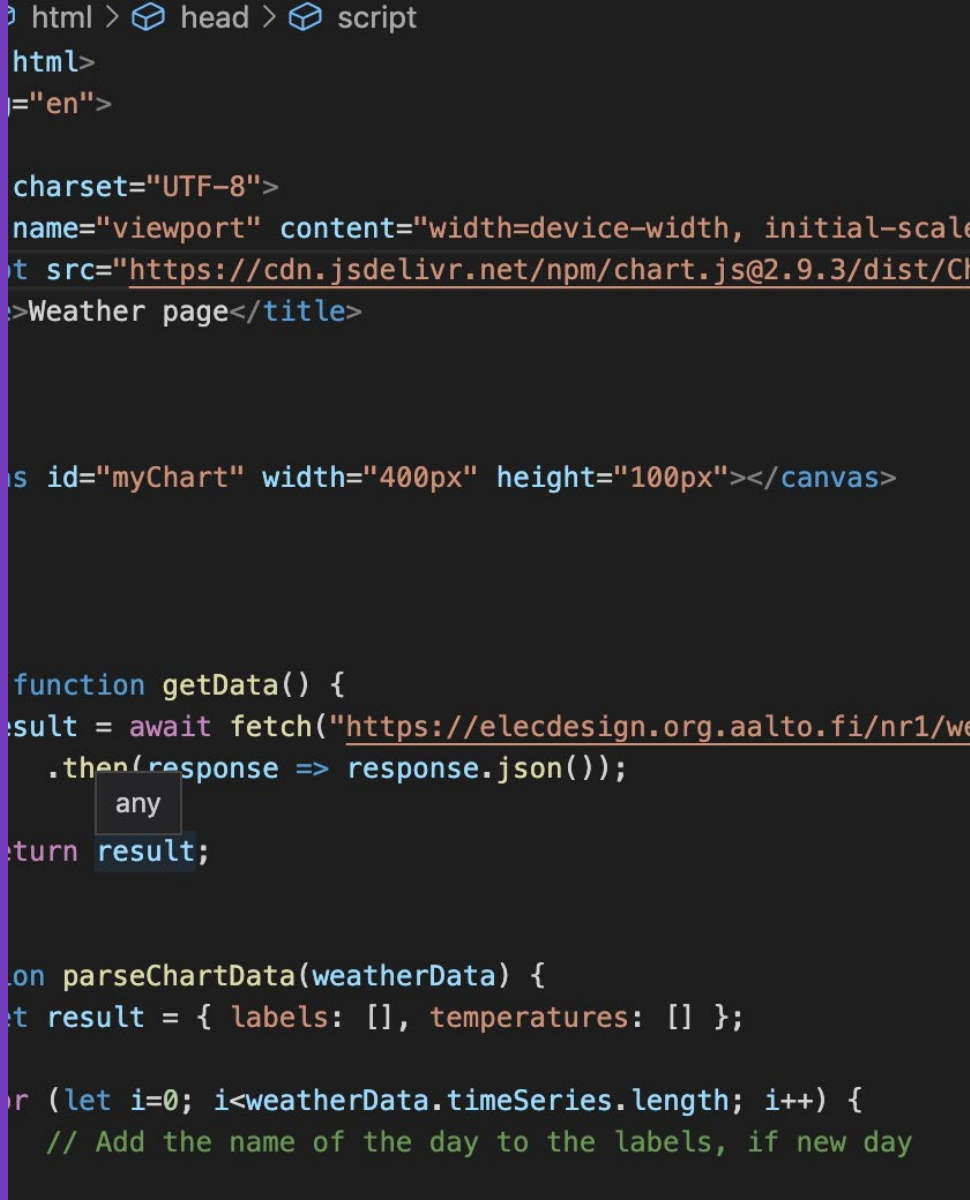
—

Skills session #3

Salu Ylirisku

11.11.2020

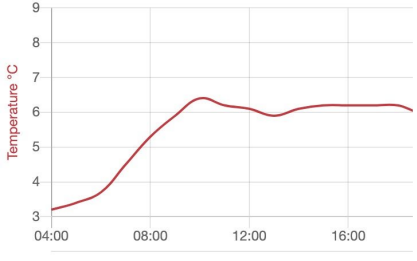
Aalto University
School of Electrical
Engineering



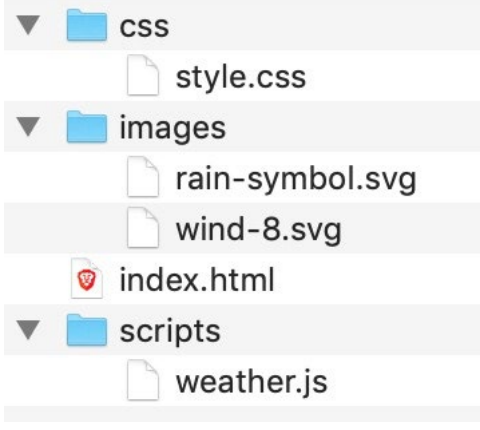
A week ago

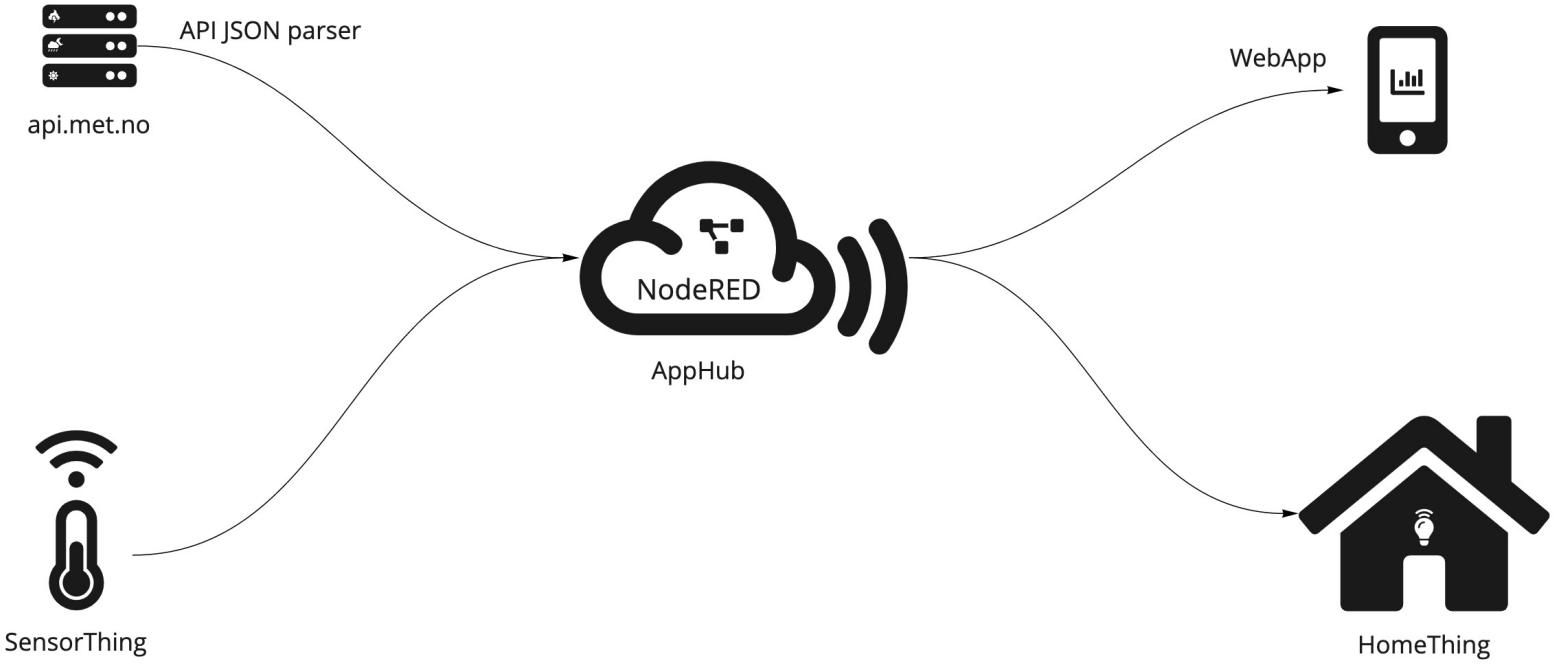
Location: Helsinki

+3.2°C



- **File hierarchy**




API JSON parser
api.met.no
WebApp
NodeRED
AppHub
SensorThing
HomeThing

Local files vs. online files

- **Last week we worked with local files**
- **These are files only visible to you on your machine**

- **The files need to be put on a web server to be shown online**

Traditionally

- **Files were simply copied to the server's file directory**
- **This is how the example WeatherApp is now put online**
  - https://elecdesign.org.aalto.fi/weatherapp/

Currently – dynamic sites!

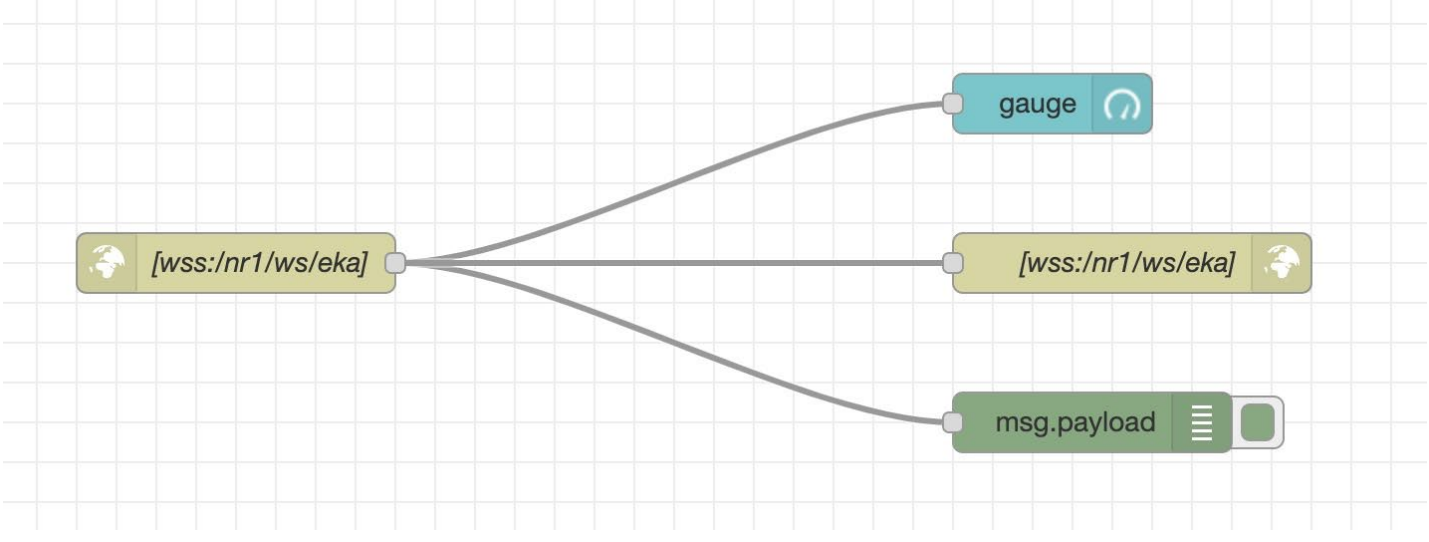
- **Majority of web ‘pages’ are today created dynamically**
- **This means**
  - What is delivered to the web browser (HTML, CSS, JS and SVGs) may be generated on the fly

JavaScript on the server side

- **JavaScript is used on the server side to create the “back-end” of dynamic web applications**
- **The most common tool is NodeJS**
  - It is the same OpenSource JavaScript engine that is used e.g. in Chrome (https://v8.dev/)
- **JavaScript takes time & effort to learn**
  - Node-RED helps to get in speed quicker
  - And provides no ceiling for the advanced programmer

Node-RED

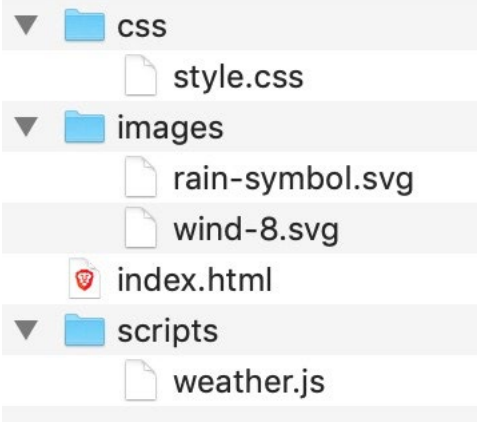
- **A visual flow-based programming tool originally developed at IBM research**

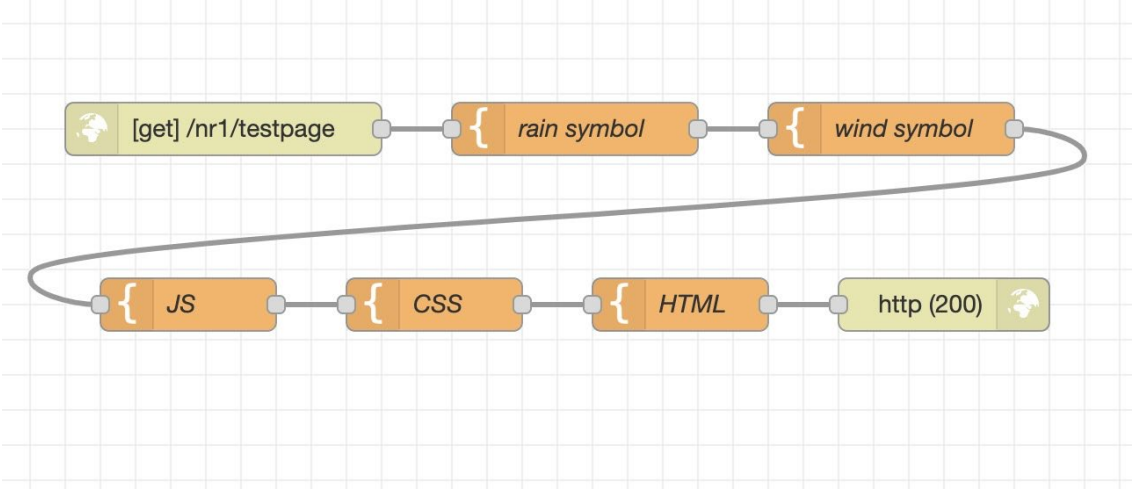


Todays exercise

- Putting last week's web page online with Node-RED
- Fetching api.met.no data from your chosen location and simplifying it for your web page

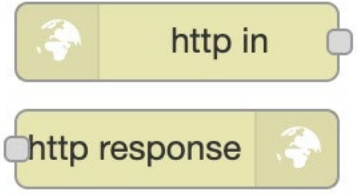
Local file structure vs. dynamic page





Nodes to use for the web page

- **HTTP in and out**



- **Template**

Demo

Try out

- **Each team has own Node-RED instance with custom username (team1, team2, etc.) and password (Salu sends individually via chat)**
- **Note: only one person should edit the nodes at a time**
  - Otherwise your changes may be over-written

api.met.no

- **Fetching api.met.no data from your chosen location and simplifying it for your web page**

- **You can get coordinates from Open Street Map**
  - Example: https://nominatim.openstreetmap.org/search?q=helsinki+finland&format=geojson

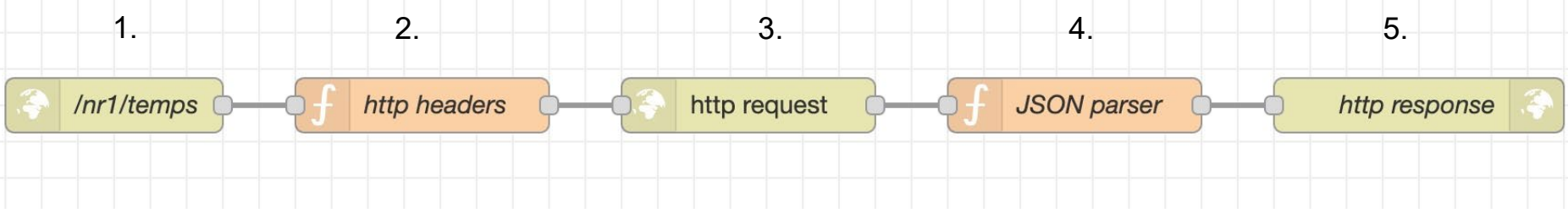
- **api.met.no**
  - https://api.met.no/weatherapi/locationforecast/2.0/documentation#!/data/get_complete

Nodes

- **http nodes**

- **function**

The node setup



1. Name for the URL
2. Setting fake HTTP headers for the API call
3. Making the API call (api.met.no)
4. Parsing the incoming data
5. Delivering the HTTP response

JSON and JavaScript Objects

string vs. object

```
{ "universities": [
    {
      "name": "Aalto",
      "location": "Espoo"
    },{
      "name": "University of Helsinki",
      "location": "Helsinki"
    }
]}
```

Accessing JS object properties:

```
universities[0]
```

```
{
  "name": "Aalto",
  "location": "Espoo"
}
```

Figuring out what is in a JSON

- http://try.jsonata.org/
- DEMO

Working with JSON objects

- Picking the wanted values (time & temp)
- Storing the values
- Passing the values on

Getting the temperature

```
properties.timeseries[0].data.instant.details.air_temperature
```