

Production Supporting Systems in Factories

ระบบสนับสนุนการผลิตในโรงงานอุตสาหกรรม

Topic

- อุปกรณ์ตรวจวัดอัจฉริยะ
- ระบบสารสนเทศ
- **Industrial Control System (ICS)**
 - SCADA (Supervisory control and data acquisition)
 - DCS (Distributed control systems)

SCADA

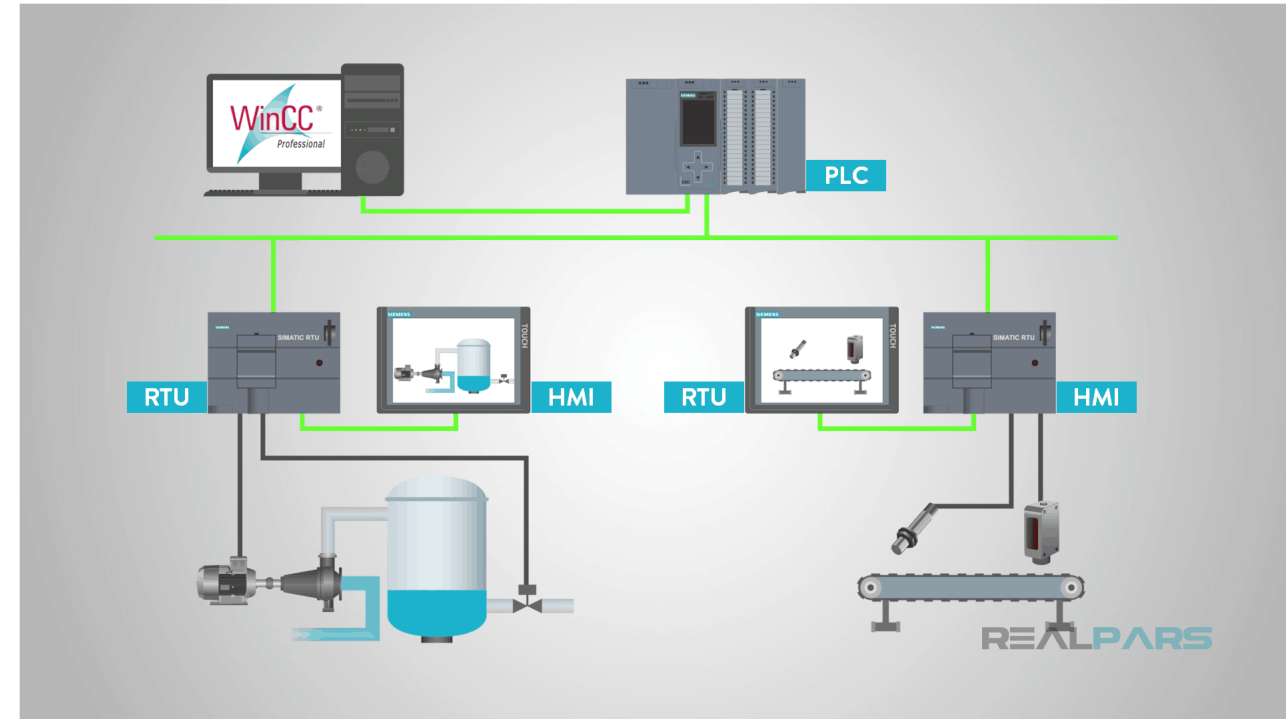
Supervisory control and data acquisition

SCADA

- A system of software and hardware elements that allows industrial organizations to:
 - **Control** industrial processes locally or at remote locations
 - Directly **interact** with devices such as sensors, valves, pumps, motors, and more through human-machine interface (HMI) software
 - **Monitor**, gather, and process real-time data
 - **Record** events into a permanent storage.

Architecture

- **PLC** (Programmable Logic Controller) and/or **RTU** (Remote Terminal Units)
- **HMI** (Human Machine Interface)
- **Field devices** - sensors and actuators
- **SCADA computer**



We will come back to this topic later.

- Now, let's learn about a control system by making a DIY "SCADA".

Project Demo

<https://prod-sup.herokuapp.com/ui>

Diagram

[Link](#)

Components

- Node-Red App running on a cloud server (Heroku Cloud)
 - Collect sensor data
 - Displaying sensor data
 - Notifying operator of the incident through Line App (on a desktop computer)
 - Writing and reading from Database (on Google Firestore)

Components (cont)

- Node-Red App running on mobile
 - Receive command to toggle a flashlight
 - Process and send accelerometer data
- Mosquitto App running on a cloud server (Google VM)
 - Receiving and broadcasting data from and to both Node-Red App.

Protocol

- HTTP (Hypertext Transfer Protocol)
 - Basically the "internet" protocol
- MQTT (MQ Telemetry Transport)
 - Leading open source protocol for connecting internet of things (IoT) and **industrial IoT (IIoT)** devices.

Comparison to SCADA components

Component	Function	SCADA Equiv.
Node-Red (Heroku)	Collect data	PLC / RTU
	Display data	SCADA computer
	Notify to Line	SCADA computer
	Write to Database	SCADA computer

Comparison to SCADA components

Component	Function	SCADA Equiv.
Node-Red (Mobile)	Control sensors	PLC / RTU
Accelerometer	Measure acceleration	Sensor
Flashlight	Light	Actuator

Enough talk. Let's get started.

Module 1-1: Setting up Node-Red

- Install [Node.js](#)
- Install [Visual Studio Code](#)

- Change default terminal in `VSCode` (Windows only)
 - Open a window
 - Select `terminal` -> `New Terminal`
 - press `ctrl` + `p`
 - Type `> Terminal: Select Default Profile` and click
 - Select `cmd`

- Install Node-Red
 - Create a new folder
 - Drag a folder into VSCode
 - Select terminal -> New Terminal
 - (Terminal) `npm init -y`
 - (Terminal) `npm install node-red`
 - Create a directory called `local`

- Configure Node-Red
 - Click at the file `package.json` to edit
 - Add this line `"start": "npx node-red -u ./local -p 1880"`

```
{  
  // ...  
  "scripts": {  
    "test": "echo \"Error: no test specified\" && exit 1",  
    // Add a new line here  
    // Don't forget to add the extra "," above.  
    "start": "npx node-red -u ./local -p 1880"  
  }  
  // ...  
}
```

- Start `Node-Red`
 - (Terminal) `npm start`
- Visit the web browser at
 - `http://127.0.0.1:1880` or
 - `http://localhost:1880`

Module 1-2: Navigating around Node-Red

Create flows to

- Show timestamp in the `debug` panel.
- Show date and time using `function` node. (See code on the other page.)
- Send continuous random numbers. (Use `Math.random()`)
- Use `switch` and `change` .

```
const payload = msg.payload;  
const date = new Date(payload);  
msg.payload = date.toDateString();  
return msg;
```

Module 1-3: HTTP `Get` Request

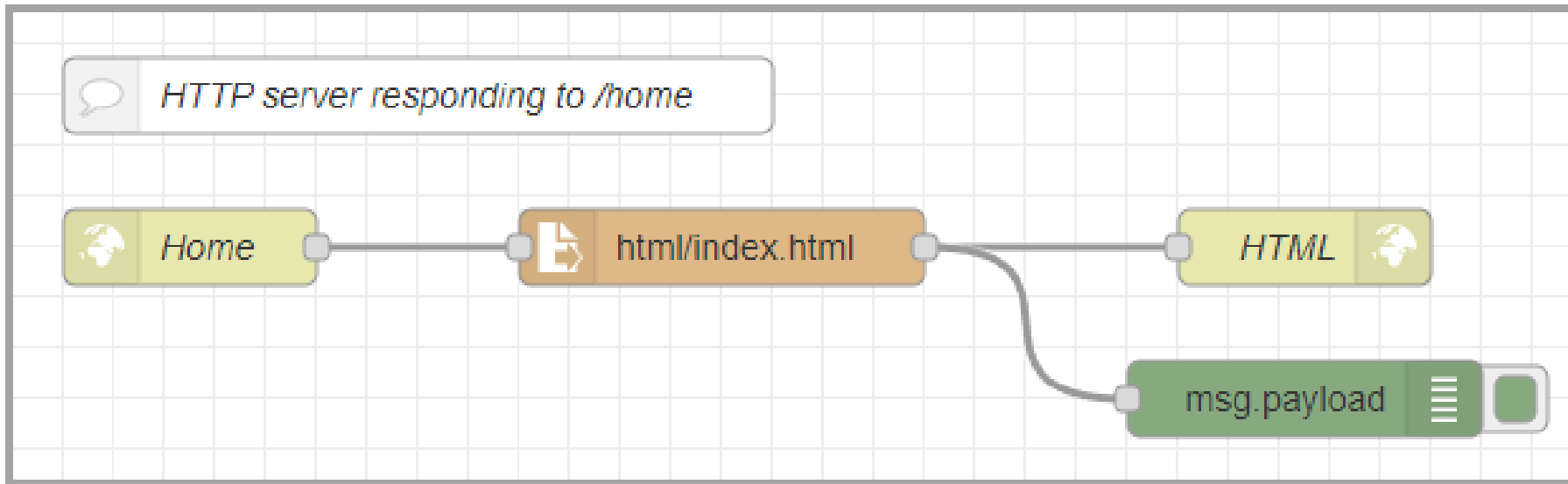
- Send a `get` request to `http://google.com` .
- Display a response to a `debug` panel.
- Write a response to `index.html`

Module 1-4: HTTP Server

- Create a server that responds to the `GET` request to `/hello`.
 - Respond with a static string.
 - Respond with a dynamic string.

```
const date = new Date();  
msg.time = date.toLocaleTimeString();  
return msg;
```

- Reponse to `/home` with an HTML document.
 - Create flow `http in`, `read file`, `http response`, `debug`



- `http in node`
 - `Method = GET`
 - `URL = home`
- `read file node`
 - `Filename = html/index.html`
 - See next page.

Edit read file node

Delete

Cancel

Done

⚙️ Properties

⚙️

📄

🔗

📄 Filename

html/index.html

➡️ Output

a single utf8 string

▼

🚩 Encoding

default

▼

🏷️ Name

Name

Tip: The filename should be an absolute path, otherwise it will be relative to the working directory of the Node-RED process.

- http response node
 - No need to do anything.


- Create a folder `html`
- Place an `index.html` in the folder
- Use web browser to the url `/home`

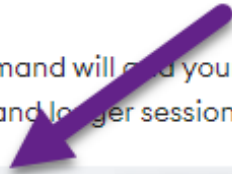
Module 1-5: Public server

- Get `ngrok`
 - Create an account at <https://ngrok.com/>
 - **Verify email.**
 - Download the program.
 - Extract the program to the project folder.
 - Copy the command below.

2. Connect your account

Running this command will add your authtoken to the default `ngrok.yml` configuration file. This will grant you access to more features and longer session times. Running tunnels will be listed on the [status page](#) of the dashboard.

```
$ ./ngrok authtoken 
```



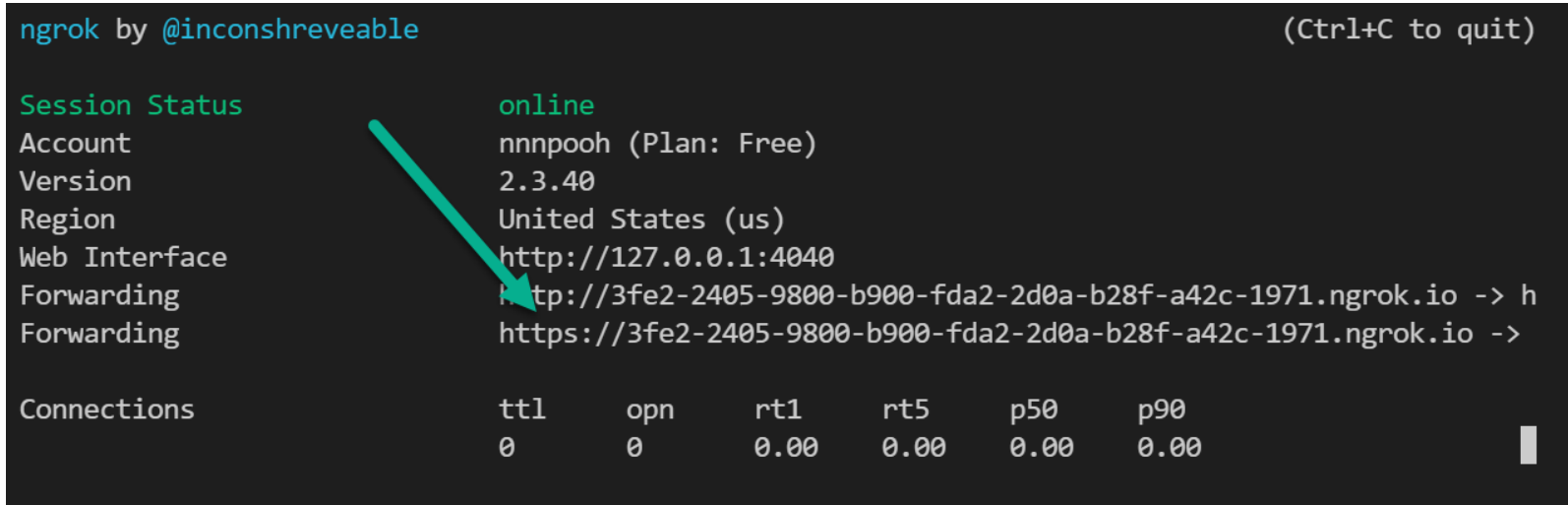
Copy

- VSCode
 - Create New Terminal
 - Run `ngrok authtoken <YOUR AUTHTOKEN>` (Remove `./` from what you just copied.)
 - Run `ngrok http 1880`
- Copy the `https` address (`ctrl` + `shift` + `c`)

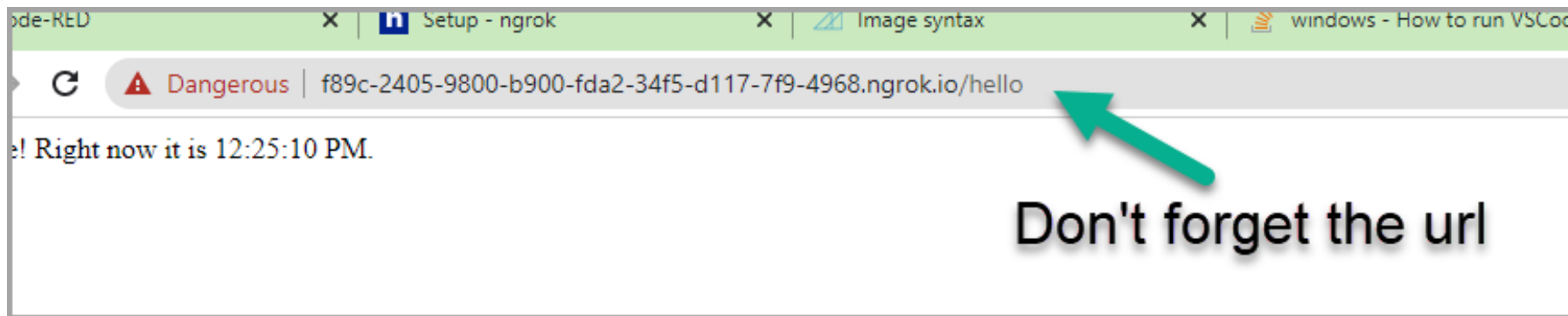
```
ngrok by @inconshreveable (Ctrl+C to quit)

Session Status      online
Account             nnnpooh (Plan: Free)
Version             2.3.40
Region              United States (us)
Web Interface        http://127.0.0.1:4040
Forwarding           http://3fe2-2405-9800-b900-fda2-2d0a-b28f-a42c-1971.ngrok.io -> h
Forwarding           https://3fe2-2405-9800-b900-fda2-2d0a-b28f-a42c-1971.ngrok.io ->

Connections         ttl    opn    rt1    rt5    p50    p90
                   0      0      0.00   0.00   0.00   0.00
```



- Paste the address in the browser.
 - Don't forget to add the url in the end.



- Try sharing this link to your friends.