

# digital.auto

## Initiators



## Standards & Open Source



## Academic Partners



# Plugin Development Hello World

*an easy getting started guide*

16-May-2023

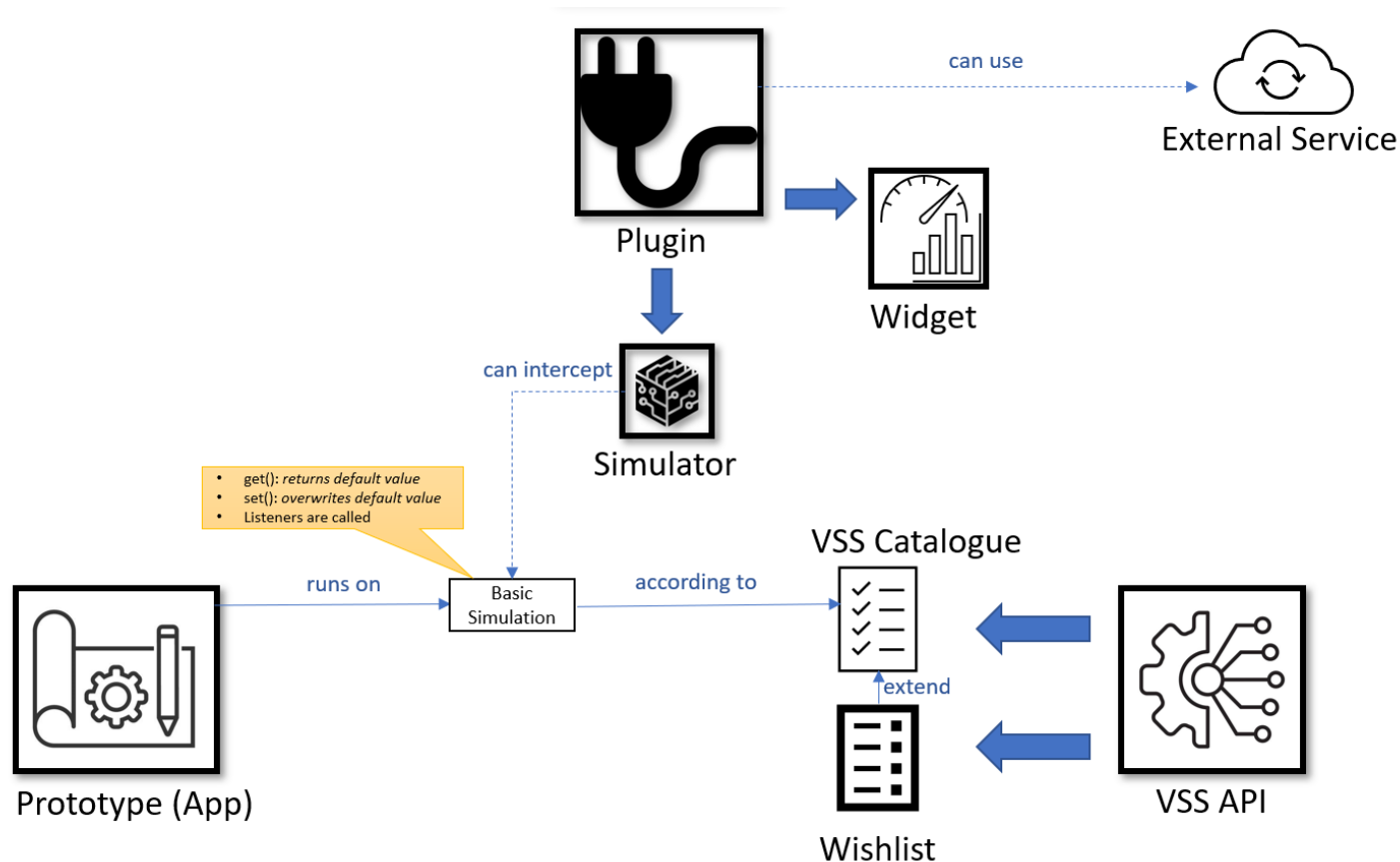
Version 0.1

# Overview

1. What is a plugin
2. Plugin template
3. Develop local plugin
4. Deploy a public plugin to Github Page

# 1. WHAT IS A PLUGIN?

- Plugin provide a mechanism for developer to add different of type of widget and logic for a model.
- Plugin is a .js file. The file be host somewhere and available to access public from internet.
- A plugin be added to the playground by using plugin URL.



## 2. PLUGIN TEMPLATE

```
const plugin = ({widgets, simulator, vehicle}) => {

  const container = document.createElement("div")
  container.setAttribute("style", `height: 100%; width: 100%;`)
  container.innerHTML = (`
    <div style="height:100px;padding: 20px; text-align: center;">
      <div style="font-size: 18px;">Hello World !!!:</div>
    </div>
  `)

  widgets.register("Helloworld",
    (box) => {
      box.injectNode(container)
    })
}

export default plugin
```

- Plugin is a javascript function with 3 parameters: widgets, simulator, vehicle
  - **widgets**: use widgets.register to register new widget. You can register multiple widgets inside plugin.
  - **simulator**: using to simulate for a VSS API.
  - **vehicle**: using to get/set a VSS API
- This is a simple plugin to show how to register a widgets, and then, get and set HeadLight value.

### 3. Develop local plugin

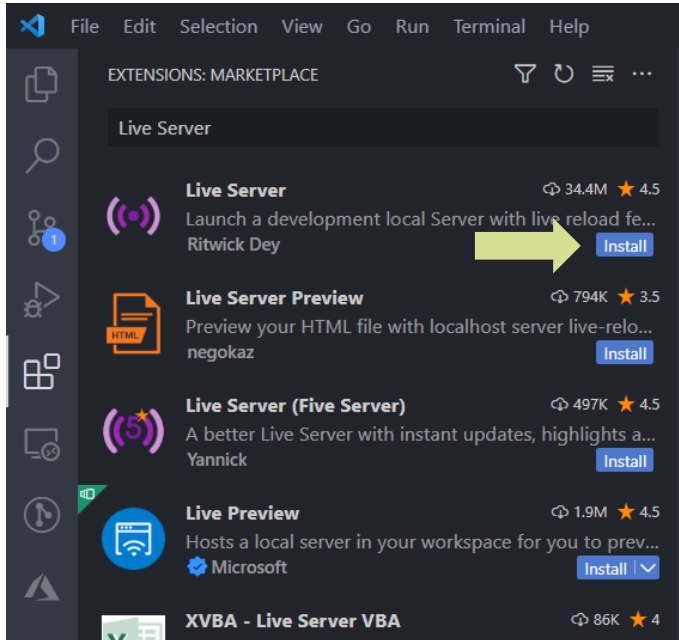
To quickly develop and test a local plugin, follow below step:

1. Create a new project with **VS Code**
2. Inside **VS Code**, install plugin **Live Server**
3. Write [plugin\_name].js file
4. Start **Live Server** plugin, you will get plugin URL: [http://localhost:5500/\[plugin\\_name\].js](http://localhost:5500/[plugin_name].js)
5. Go to digital playground, add a plugin with above URL
6. Make a prototype using above plugin, then test the plugin
7. If plugin working not correctly, edit the [plugin\_name].js file in Vscode, save, and reload digital.playground to test again.

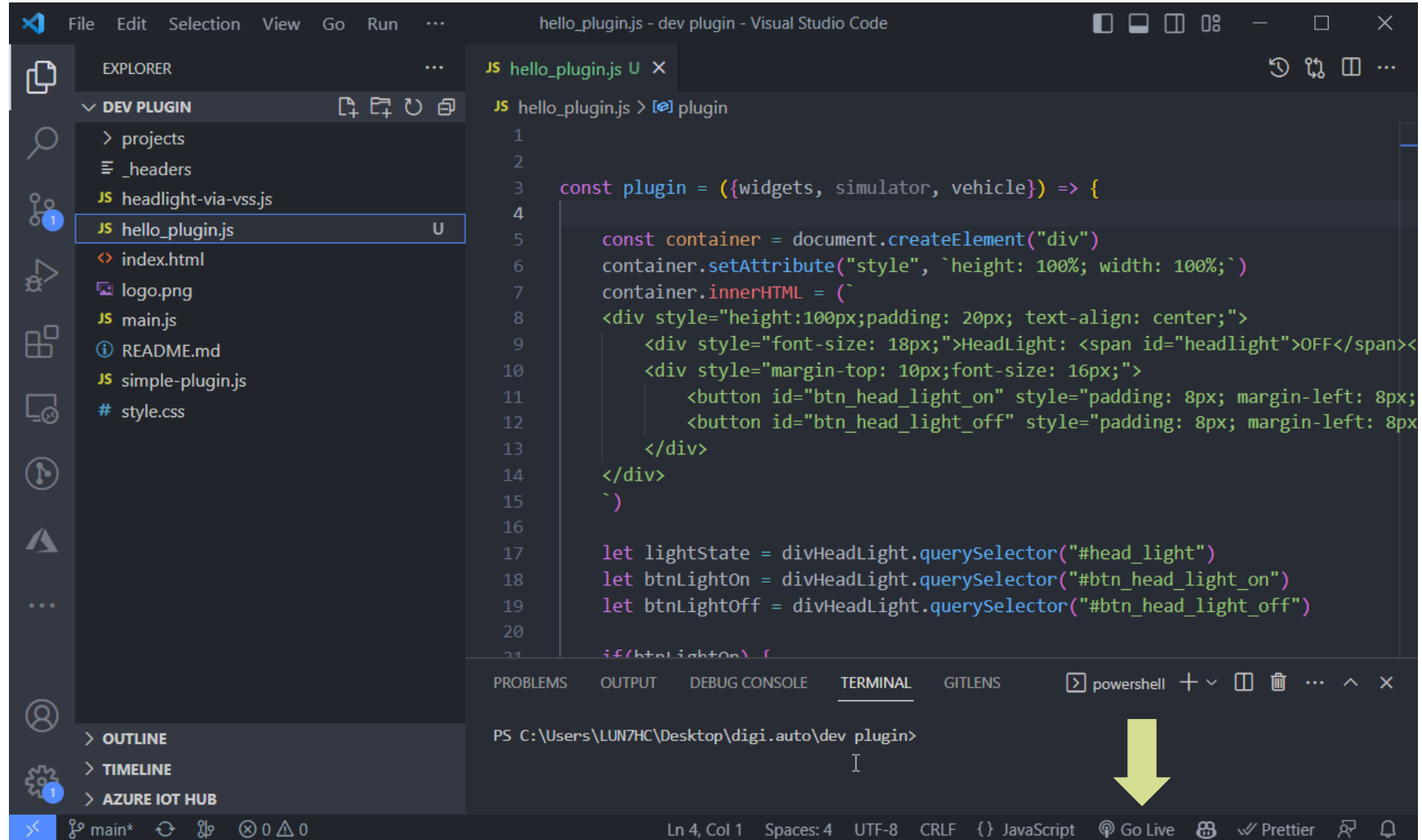
Enjoy!

# Live Server demo

## 1. Install Live Server:



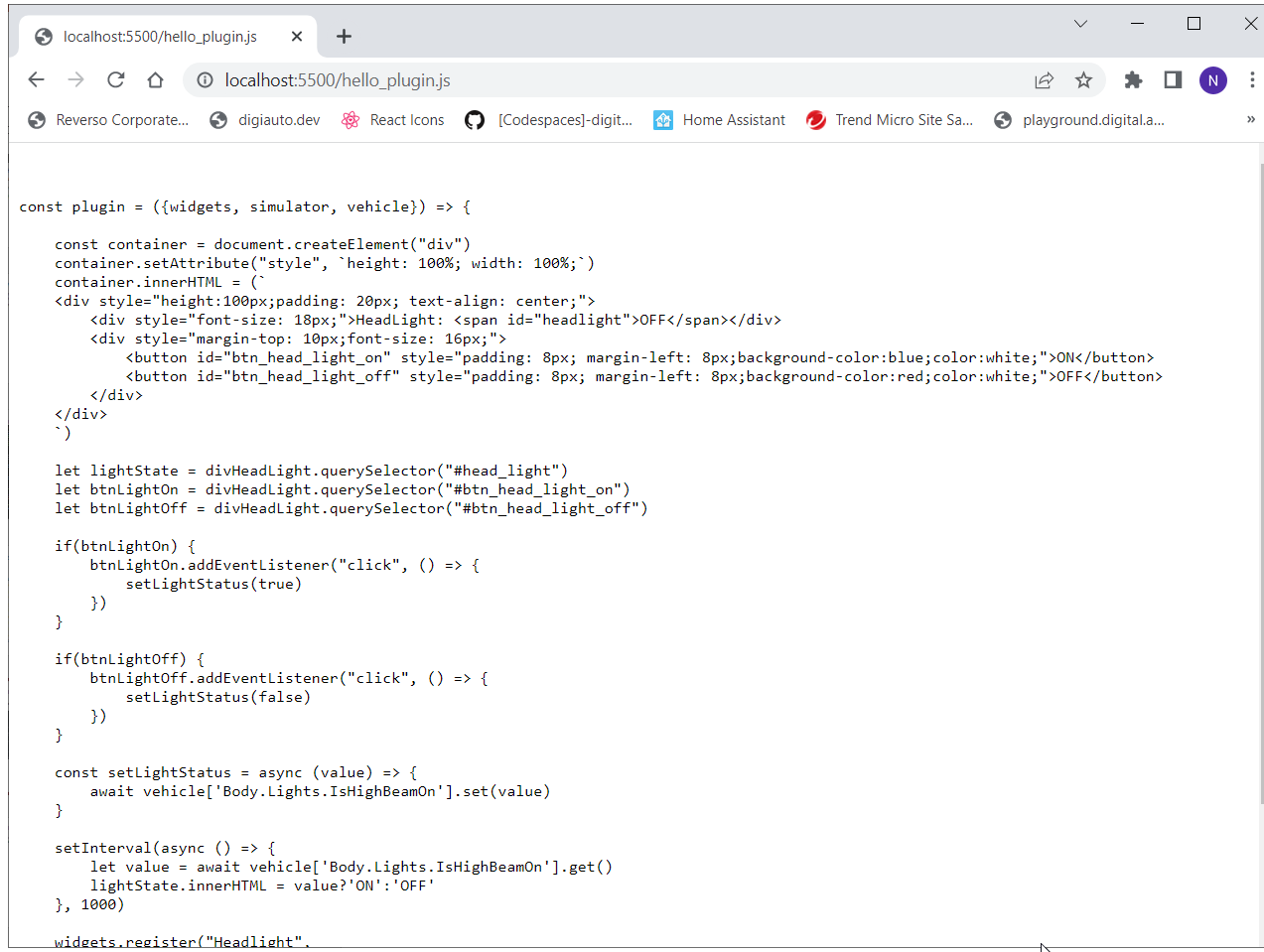
## 2. Click **Go Live** at bottom to start server



# Live Server demo (cont.)

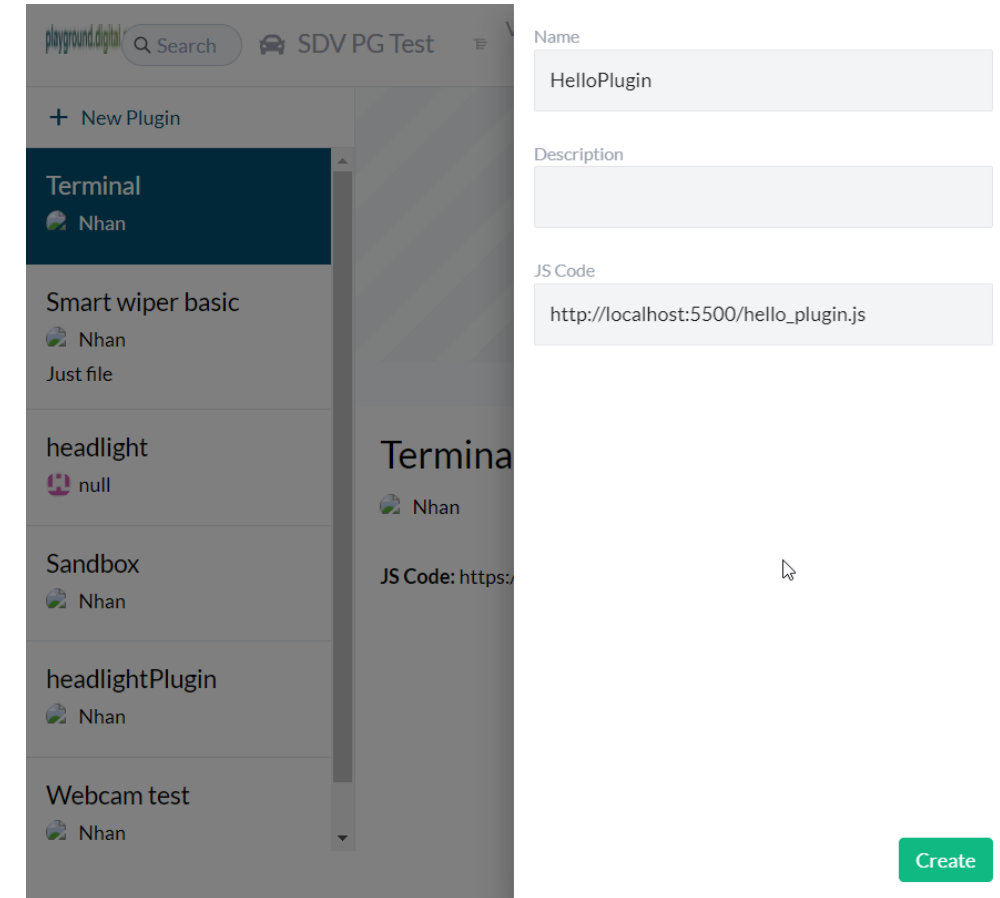
3. JS URL is ready to use, test it by Chrome

[http://localhost:5500/hello\\_plugin.js](http://localhost:5500/hello_plugin.js)

A screenshot of a web browser window. The address bar shows 'localhost:5500/hello\_plugin.js'. The page content is a JavaScript code snippet that defines a plugin for a headlight simulation. The code includes HTML templates for the headlight and buttons to turn it on/off, and logic to interact with a vehicle's body lights.

```
const plugin = ({widgets, simulator, vehicle}) => {  
  const container = document.createElement("div")  
  container.setAttribute("style", `height: 100%; width: 100%;`)  
  container.innerHTML = (`  
    <div style="height:100px;padding: 20px; text-align: center;">  
      <div style="font-size: 18px;">HeadLight: <span id="headlight">OFF</span></div>  
      <div style="margin-top: 10px;font-size: 16px;">  
        <button id="btn_head_light_on" style="padding: 8px; margin-left: 8px;background-color:blue;color:white;">ON</button>  
        <button id="btn_head_light_off" style="padding: 8px; margin-left: 8px;background-color:red;color:white;">OFF</button>  
      </div>  
    </div>  
  `)  
  let lightState = divHeadLight.querySelector("#head_light")  
  let btnLightOn = divHeadLight.querySelector("#btn_head_light_on")  
  let btnLightOff = divHeadLight.querySelector("#btn_head_light_off")  
  
  if(btnLightOn) {  
    btnLightOn.addEventListener("click", () => {  
      setLightStatus(true)  
    })  
  }  
  
  if(btnLightOff) {  
    btnLightOff.addEventListener("click", () => {  
      setLightStatus(false)  
    })  
  }  
  
  const setLightStatus = async (value) => {  
    await vehicle['Body.Lights.IsHighBeamOn'].set(value)  
  }  
  
  setInterval(async () => {  
    let value = await vehicle['Body.Lights.IsHighBeamOn'].get()  
    lightState.innerHTML = value?'ON':'OFF'  
  }, 1000)  
  
  widgets.register("Headlight",
```

4. Add new plugin to playground



5. From this point, you can change hello\_plugin.js in VS Code, then save. And then reload the playground, new change will be applied automatically. Enjoy!



# 4. DEPLOY A PUBLIC PLUGIN TO GITHUB PAGE

- To quickly public your plugin, using GitHub Page
  - Create a **public repository** in GitHub with this name: [GitHubUsername].github.io
  - Push your plugin code in last step to this repository.

Voila! now you have plugin ready to use at URL:

**[https:// \[GitHubUsername\].github.io/\[plugin\\_name\].js](https://[GitHubUsername].github.io/[plugin_name].js)**

## 1. Create a public repository

❶ and ❷ must be the same, it is your username

<> Start writing code

**Start a new repository**

A repository contains all of your project's files, revision history, and collaborator discussion.

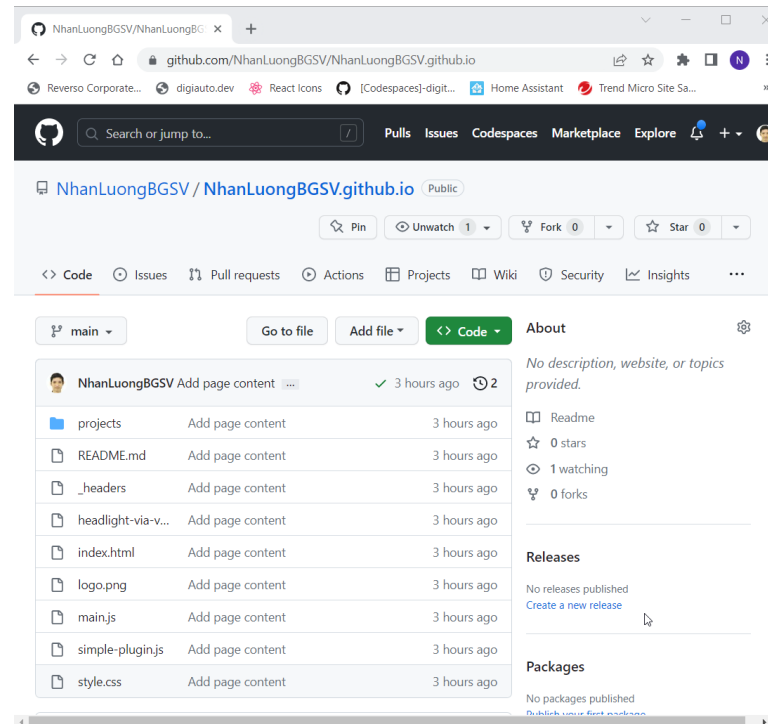
NhanLuongBGSV / NhanLuongBGSV.github.io

☒ **Public**  
Anyone on the internet can see this repository

☐ **Private**  
You choose who can see and commit to this repository

Create a new repository

## 2. Push your code to repository



## 3. Your plugin now become public

