Install VS code (*Note : JDK 8 or JDK 11 are required*) =>   
<https://trailhead.salesforce.com/content/learn/projects/quick-start-lightning-web-components/set-up-visual-studio-code>

Install SFDX =>   
<https://trailhead.salesforce.com/content/learn/projects/quick-start-lightning-web-components/set-up-salesforce-dx>

Few SFDX CLI commands

1. To install sfdx updates => sfdx plugins:install salesforcedx@latest
2. To verify if sfdx is installed properly => sfdx force
3. To view list of orgs connected => sfdx force:org:list
4. To create new scratch org => sfdx force:org:create
5. To open scratch org => sfdx force:org:open --targetusername *uName*
6. To see all limits => sfdx force:limits:api:display --targeusername *uName*
7. To open default scratch org => sfdx force:org:open
8. To delete scratch org => sfdx force:org:delete --targetusername *uName*

**What is Scratch Org –**   
A scratch org is a dedicated, configurable, and short-term Salesforce environment. Scratch orgs drive developer productivity and collaboration during the development process, and facilitate automated testing and continuous integration. You can use the CLI or Salesforce Extensions for VS Code to open your scratch org in a browser without logging in.

***Hello World in a Scratch Org = >***

Let’s create a Hello World component and use Lightning App Builder to add it to a Lightning page. Then, we’ll view it in a scratch org.

Create a Salesforce DX Project -   
In VS Code, press Command + Shift P, enter sfdx, and select **SFDX: Create Project**. For project name, enter lwc-hello-world.

Authorize Your Dev Hub Org -   
Dev Hub is a central location for Salesforce DX because it allows you to create, delete, and manage your Salesforce scratch orgs.   
Before you can create a scratch org, authorize your Dev Hub org. Running this command opens a browser to the Salesforce login page. Enter your Dev Hub username and password. Authorize your Dev Hub only the first time you work on a project.  
In VS Code, press Command + Shift P, enter sfdx, and select **SFDX: Authorize a Dev Hub**.

Create a Default Scratch Org -   
In VS Code, press Command + Shift P, enter sfdx, and select **SFDX: Create a Default Scratch Org**. Accept the default values.

Create a Lightning Web Component -   
Create Lightning web components in the force-app/main/default/lwc folder.  
In VS Code, press Command + Shift P, enter sfdx, and select **SFDX: Create Lightning Web Component**. For desired directory, accept the default. For filename, enter helloWorld.

Use camel case to name your Lightning web components (helloWorld) because it maps to kebab-case in HTML (<c-hello-world>).

Add this code to the helloWorld.js file.

import { LightningElement**, api** } from 'lwc';

export default class HelloWorld extends LightningElement {

**@api name;**

}

The [@api decorator](https://developer.salesforce.com/docs/component-library/documentation/lwc/reference_decorators.html) makes the name property public.

Add this code to the helloWorld.html file.

<template>

**<lightning-card title="HelloWorld" icon-name="custom:custom14">**

**<div class="slds-card\_\_body slds-card\_\_body\_inner">**

**Hello, {name}!**

**</div>**

**</lightning-card>**

</template>

The [{} syntax](https://developer.salesforce.com/docs/component-library/documentation/lwc/js_props_getter.html) binds the name property in the HTML template to the name property in the JavaScript class. The other markup gives the text some padding and wraps it in the lightning-card base Lightning component to make it look better.

Configure the Component for Lightning App Builder

Configure the helloWorld component for Lightning App Builder so that we can add it to a Lightning page in our scratch org.

Open helloWorld.js-meta.xml and add the code in bold. To use the component in Lightning App Builder, set <isExposed> to true. The <target> lets us add the component to an app page. The <targetConfigs> section lets us set the component’s name property in Lightning App Builder.

*<?xml version="1.0" encoding="UTF-8"?>*

<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata" fqn="anotherComponent">

<apiVersion>45.0</apiVersion>

<isExposed>**true**</isExposed>

**<masterLabel>Hello World</masterLabel>**

**<description>Add a classic greeting to any page.</description>**

**<targets>**

**<target>lightning\_\_AppPage</target>**

**</targets>**

**<targetConfigs>**

**<targetConfig targets="lightning\_\_AppPage">**

**<property name="name" type="String" label="Name" placeholder="World" description="Enter the name of the person to greet."/>**

**</targetConfig>**

**</targetConfigs>**

</LightningComponentBundle>

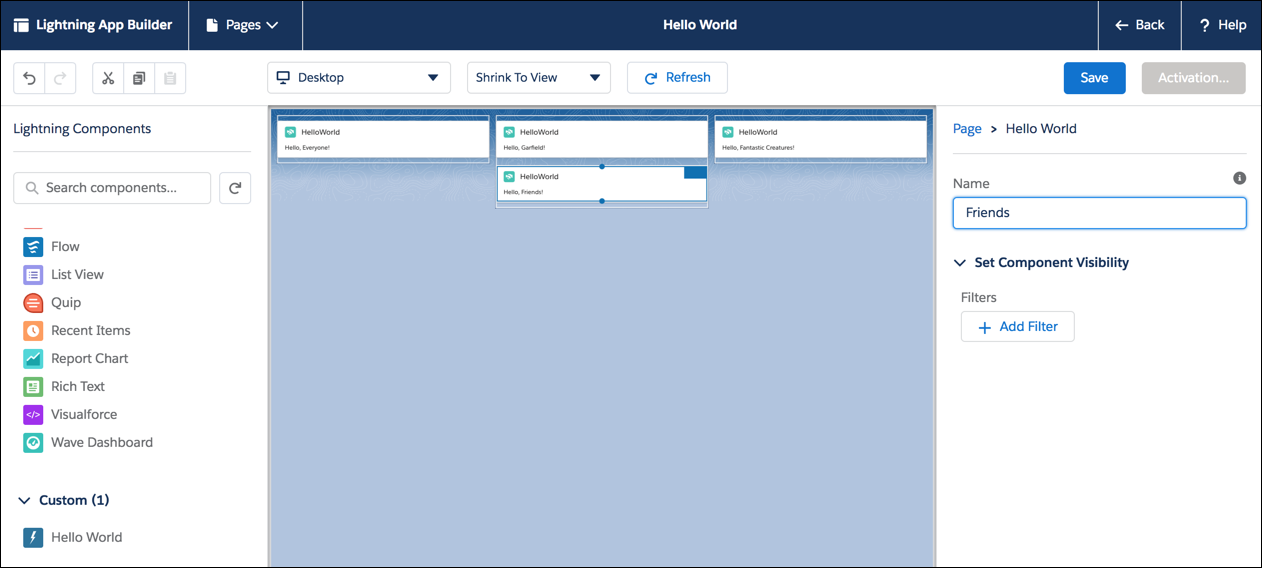
You can also add [targets](https://developer.salesforce.com/docs/component-library/documentation/lwc/reference_configuration_tags.html) for the Lightning Experience home page, record pages, and Community Builder.

Push Source to the Scratch Org -   
In VS Code, press Command + Shift P, enter sfdx, and select **SFDX: Push Source to Default Scratch Org**.

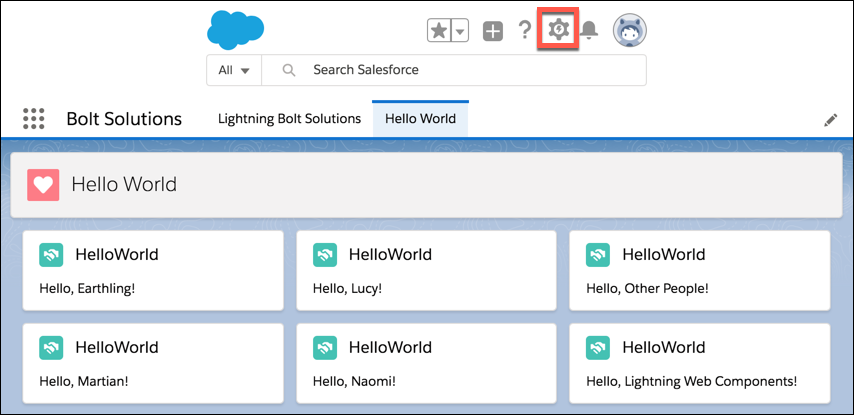
Open the Scratch Org -   
In VS Code, press Command + Shift P, enter sfdx, and select **SFDX: Open Default Scratch Org**.

Add the Component to a Lightning Page

1. In the scratch org, in Setup, enter bui and click Lightning App Builder.
2. To create a Lightning page, click New.
3. Select App Page and click Next.
4. Enter the label Hello World.
5. Select Three Regions and click Finish.
6. Drag the Hello World component from the Custom list of components to a region on the page.



1. Select the component and enter a Name.
2. Drag more Hello World components to the page, and set a different name for each component.
3. When your page is complete, click Save and Activate.
4. On the Activation page, select an icon, then click Lightning Experience, and add the page to the Lightning Bolt app. Click Save.
5. To exit Lightning App Builder, click Back.
6. From the App Picker, click Bolt Solutions.
7. Click the Hello World tab to see your page and all its greetings.



To return to Lightning App Builder to edit the page, click the gear and select Edit Page.

When you edit code, push it to the scratch org and hard refresh the browser.

**To manage scratch orgs from Salesforce UI -**

1. Go to App Launcher, search and select “Active Scratch Org”. List of all active scratch orgs will be displayed.
2. To delete an active scratch org from the Active Scratch Org list view, choose Delete from the dropdown.
3. To view the requests that created the scratch orgs, select Scratch Org Info from the App Launcher.

Trailhead link to learn more on SFDX => <https://trailhead.salesforce.com/en/content/learn/trails/sfdx_get_started>   
Developer Guide for Scratch org using SFDX => <https://developer.salesforce.com/docs/component-library/documentation/lwc/lwc.get_started_sfdx_hello_world>

Scratch org definition file (project-scratch-def.json) => <https://developer.salesforce.com/docs/atlas.en-us.sfdx_dev.meta/sfdx_dev/sfdx_dev_scratch_orgs_def_file.htm>

Scratch Org Trailhead <https://trailhead.salesforce.com/content/learn/modules/sfdx_app_dev>

Push code from scratch org to Dev Hub - <https://wipdeveloper.com/salesforce-dx-push-changes-to-dev-hub/>

Deployments can also be done by normal package.xml file using all different tools the way other metadata is deployed as MetaData API supports LWC.

**Connect to org directly and develop code without scratch org the way we used to do with editors earlier. Such orgs are called as non-source-tracked orgs.**

1. Create new project => SFDX: Create Project with Manifest
2. Select standard template then provide name of the project and then click enter.
3. Pick existing org if already present or authorize new org if not present or doing it for first time.
4. To authorize new org => SFDX: Authorize an Org
5. Modify manifest if required
6. Get existing metadata from source org => SFDX: Retrieve Source in Manifest from Org
7. Folder retrieve and file retrieve can be done using SFDX: Retrieve Source from Org or SFDX: Retrieve This Source from Org respectively.
8. Similarly deploy commands can be used to deploy.

Additionally,

1. Delete default org from VS code => SFDX: Delete Default Org
2. Delete all expired orgs from VS code => SFDX: Remove All Expired Orgs