Day 10:

Topic: LWC

Milestones: Setting up LWC

Salesforce CLI

Like many other programming languages and models, Salesforce includes a command-line interface (CLI). If you've ever used npm, yarn, gradle, or maven, Salesforce CLI will seem familiar to you—just tailor-made for Salesforce development tasks (and if those acronyms look like alphabet soup to you, that's fine, too).

Salesforce CLI allows you to interact with your Salesforce environments in many ways, like retrieving or pushing code or interacting with data. The CLI consists of several plugins. These plugins provide important specific functionality. For example, the salesforcedx plugin provides the ability to interact with Salesforce orgs and their data.

Complete installation instructions.

1.Install the CLI from https://developer.salesforce.com/tools/sfdxcli.
2.Confirm the CLI is properly installed and on the latest version by running the following command from the command line.
sfdx update
You should see output like **sfdx-cli: Updating CLI....**

Visual Studio Code

Visual Studio Code is the go-to code editor for Salesforce developers. It is free, open-source, and available for Windows, Linux, and macOS. Visual Studio Code is a well-established IDE among web developers. Now it's also an effective IDE for building Lightning web components, and Salesforce provides free extensions for Visual Studio Code that simplify your development experience even more.

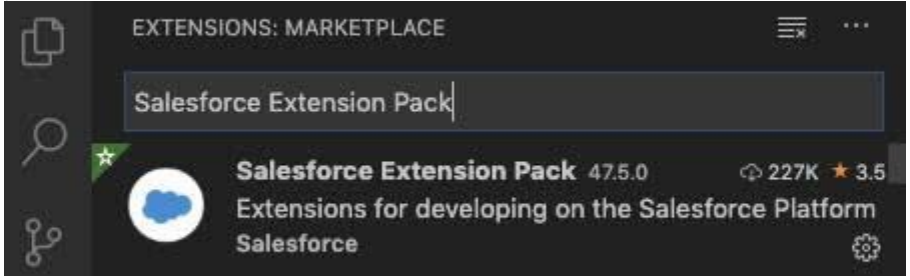
Follow these instructions to install Visual Studio Code.

1.Download and install the latest version of Visual Studio Code for your operating system. If you already have Visual Studio Code installed, there's no need to reinstall it.

2.Launch Visual Studio Code.

3.Click the ⊞ icon for Extensions in the sidebar.

4.Search for Salesforce Extension Pack and click Install. If you already have it installed, then you just need to click on the Reload button.


EXTENSIONS: MARKETPLACE
Salesforce Extension Pack
Salesforce Extension Pack 47.5.0
227K ★ 3.5
Extensions for developing on the Salesforce Platform
Salesforce

5.Press Command + Shift + P on macOS or Ctrl + Shift + P on Windows or Linux to reveal the command palette. In the command palette, type sfdx to display an initial list of available commands.

Done! You've installed all the tools you need for developing your first Lightning web component.

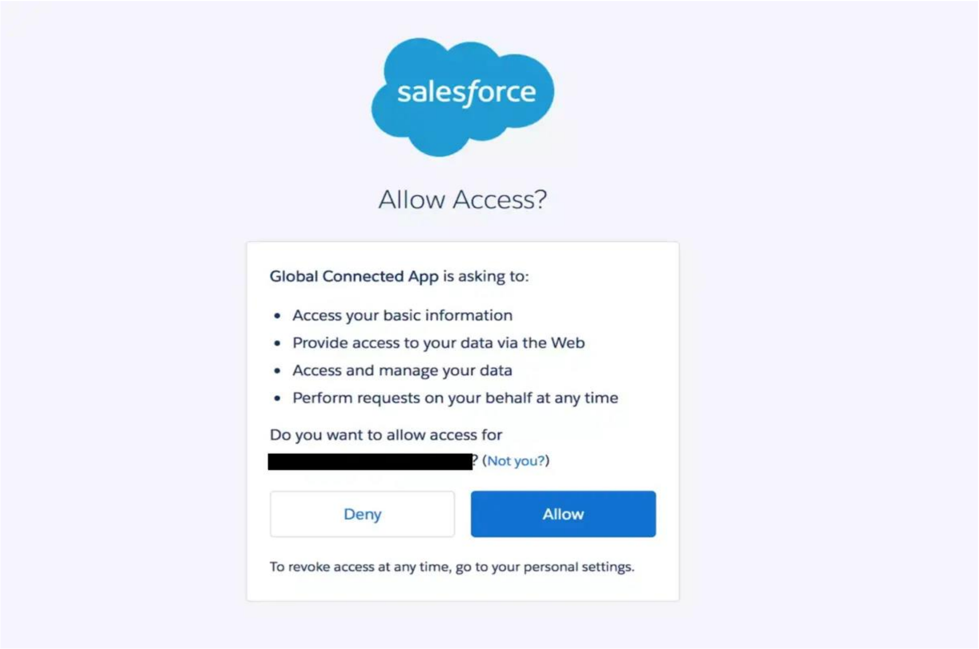
Create a Salesforce DX Project

The basic foundation for interacting with an org using Salesforce CLI is a Salesforce DX project. A project consists of several local configuration files, as well as the code you want to deploy. In Salesforce terms we call this code metadata, which is the foundation of the Salesforce Platform. If you're new to Salesforce, check out Platform Development Basics to learn more about our metadata-driven approach.

1. Open **Visual Studio Code**.
2. Press **Command + Shift + P** on macOS or **Ctrl + Shift + P** on Windows or Linux, then type **create project**. Select **SFDX: Create Project**, and press **Enter**.
3. Leave the default project type selection **Standard** as is, and press **Enter**.
4. Enter **trailhead** as project name, and press **Enter**.
5. Choose a directory on your local machine where the project will be stored. Click **Create Project**.

Authorize Your Dev Hub

1. In **Visual Studio Code**, press **Command + Shift + P** on macOS or **Ctrl + Shift + P** on Windows or Linux.
2. Type **sfdx**.
3. Select **SFDX: Authorize a Dev Hub**.
4. Log in using your Dev Hub org credentials.
5. Click **Allow**.



Allow Access?

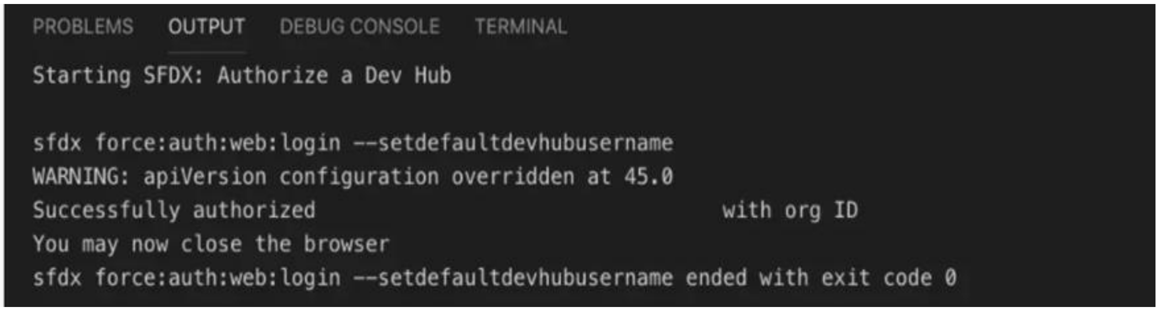
Global Connected App is asking to:

- Access your basic information
- Provide access to your data via the Web
- Access and manage your data
- Perform requests on your behalf at any time

Do you want to allow access for [redacted]? (Not you?)

To revoke access at any time, go to your personal settings.

6. After you authenticate in the browser, the CLI remembers your Dev Hub credentials. The success message should look like this:



```
Starting SFDX: Authorize a Dev Hub

sfdx force:auth:web:login --setdefaultdevhubusername
WARNING: apiVersion configuration overridden at 45.0
Successfully authorized                                   with org ID
You may now close the browser
sfdx force:auth:web:login --setdefaultdevhubusername ended with exit code 0
```

Create a Scratch Org

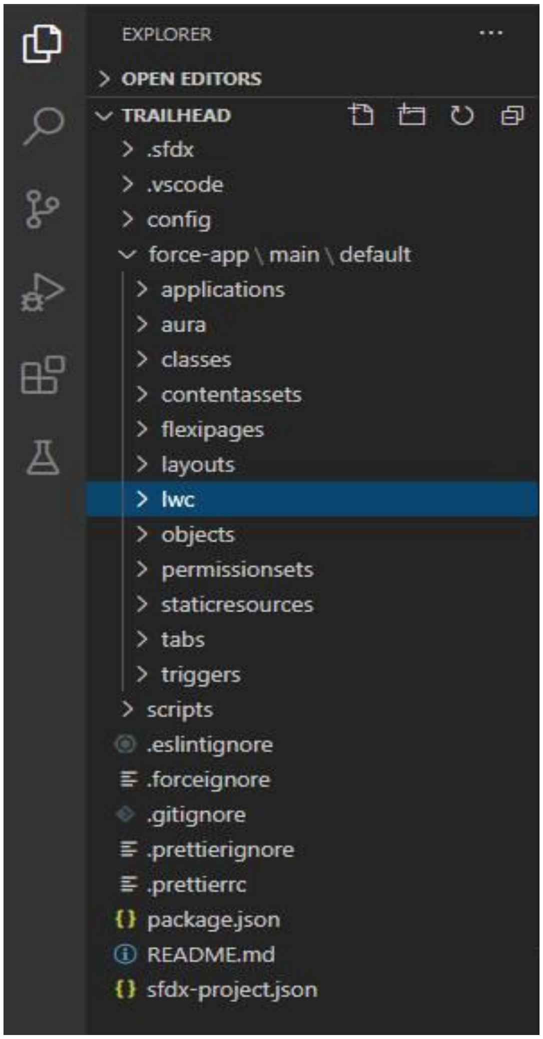
1. In Visual Studio Code, press **Command + Shift + P** on macOS or **Ctrl + Shift + P** on Windows or Linux.
2. Type **sfdx**.
3. Select **SFDX: Create a Default Scratch Org...**.
4. Press **Enter** to accept the default project-scratch-def.json.
5. Press **Enter** to accept the default trailhead scratch org alias.
6. Press **Enter** to accept the default 7 days scratch org duration.
7. Be patient, creating a scratch org can take a minute. The success message should look like this in the output panel of VS Code:

```
17:18:11.779 sfdx force:org:create -f config\project-scratch-def.json --setalias trailhead --durationdays 7 --setdefaultusername --json --loglevel fatal ended with exit code 0
```

Now you’re all set to develop your first Lightning web component

Create a Lightning Web Component

Creating a Lightning web component is a straightforward process. And Salesforce CLI already created a project structure that helps make getting started even easier.
The folder structure looks like this:



```
EXPLORER ...
> OPEN EDITORS
v TRAILHEAD
  > .sfdx
  > .vscode
  > config
  v force-app \ main \ default
    > applications
    > aura
    > classes
    > contentassets
    > flexipages
    > layouts
    > lwc
    > objects
    > permissionsets
    > staticresources
    > tabs
    > triggers
  > scripts
  .eslintignore
  .forceignore
  .gitignore
  .prettierignore
  .prettierrc
  package.json
  README.md
  sfdx-project.json
```

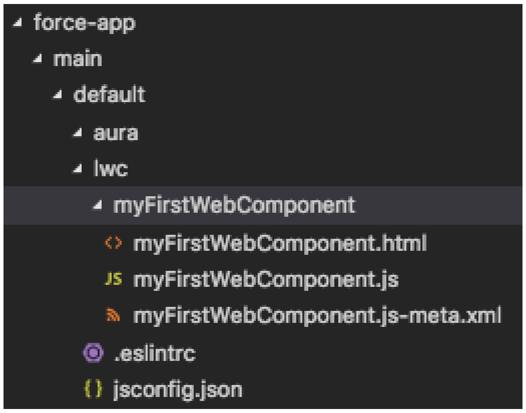
The project we created has a special folder, force-app/main/default. This folder, called a package directory, contains all the metadata of your current Salesforce DX project. Because Lightning web components are also metadata, they are stored in a subfolder named lwc. In the next step, we add a Lightning web component to this folder.

We can use Visual Studio Code for creating a Lightning web component, just as we did to create the Salesforce DX project. Or we can use Salesforce CLI directly.

1.Open Visual Studio Code.
2.Press Command + Shift + P on macOS or Ctrl + Shift + P on Windows or Linux, then type focus terminal. Press Enter.
3.Enter

```
sfdx force:lightning:component:create -n myFirstWebComponent -d force-app/main/default/lwc --type lwc
```

, and confirm with Enter.
This creates the needed files for your first Lightning web component.



```
force-app
  main
    default
      aura
      lwc
        myFirstWebComponent
          myFirstWebComponent.html
          myFirstWebComponent.js
          myFirstWebComponent.js-meta.xml
        .eslintrc
        jsconfig.json
```

Adding Code and Metadata to Your First Lightning Web Component

1. Open the new subfolder for **myFirstWebComponent**, which we just created in the lwc subfolder
2. Click **myFirstWebComponent.js-meta.xml**.
3. Replace the contents of the XML file with this XML markup

```
<?xml version="1.0" encoding="UTF-8"?>
<LightningComponentBundle xmlns="http://soap.sforce.com/2006/04/metadata">
   <apiVersion>51.0</apiVersion>
   <isExposed>true</isExposed>
   <targets>
       <target>lightning__AppPage</target>
       <target>lightning__RecordPage</target>
       <target>lightning__HomePage</target>
   </targets>
</LightningComponentBundle>
```

4. Press **CMD + S** on macOS, or **CTRL + S** on Windows or Linux, to save the file.

Next we’re updating the JavaScript file of your Lightning web component.

1. In Visual Studio Code click **myFirstWebComponent.js**.
2. Paste this

```
import { LightningElement } from 'lwc';
export default class MyFirstWebComponent extends LightningElement {
   @track
   contacts = [
      {
         Id: 1,
         Name: 'Amy Taylor',
         Title: 'VP of Engineering',
      },
      {
         Id: 2,
         Name: 'Michael Jones',
         Title: 'VP of Sales',
      },
      {
         Id: 3,
         Name: 'Jennifer Wu',
         Title: 'CEO',
      },
   ];
}
```

3.Press CMD + S on macOS, or CTRL + S on Windows or Linux, to save the file.
After you save the file you'll immediately notice a few things.

- The annotated word @track is underlined with a red squiggly line.
- The JavaScript file color in the explorer changed to red, and has a 2 next to it.

HTML markup

1. In Visual Studio Code click **myFirstWebComponent.html**.
2. Insert this markup within the existing <template></template> tags

```
<lightning-card title="ContactInformation" icon-name="custom:custom14">
  <div class="slds-m-around_medium">
    <template for:each={} for:item="contact">
      <div>
        {contact.Name}, {contact.Title}
      </div>
    </template>
  </div>
</lightning-card>
```

3.Press **CMD + S** on macOS, or **CTRL + S** on Windows or Linux, to save the file.

Deploy and Configure Your New Lightning Web Component

1. Open **Visual Studio Code**.
2. Press **Command + Shift + P** on macOS or **Ctrl + Shift + P** on Windows or Linux, then type **focus terminal**. Press **Enter**.
3. Enter this command to deploy the metadata to your org:
   ```
   sfdx force:source:push
   ```
4. Press **Enter**