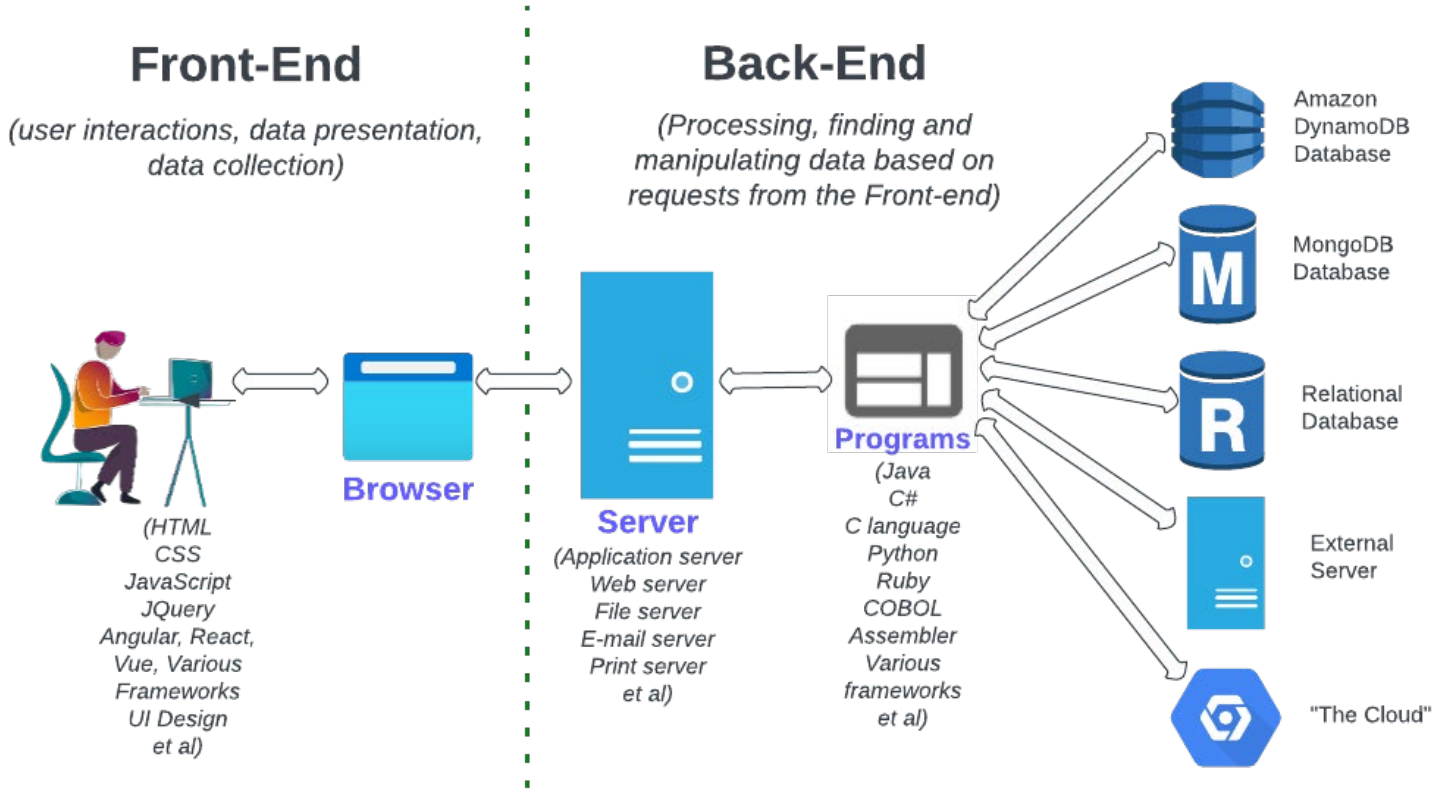
What is Angular?

Angular is an open source JavaScript framework for building single page web applications. Angular applications are built with HTML, CSS, and TypeScript and they run in the browser.



Verify and Installation *(if necessary)*

Angular updates about every 6 months or so. Most updates are designed not to break old code as it's a fast cycle, but sometimes a new version has major changes, so be sure to keep up to date.

To verify an Angular installation and install, if necessary:

1. Open a GitBash Session.
2. Run the following command to verify Angular is installed: `ng version`

Expected output *(your version numbers may be different, that should be OK)*:

```
     _                      _                 ____ _     ___
    / \   _ __   __ _ _   _| | __ _ _ __     / ___| |   |_ _|
   / △ \ | '_ \ / _` | | | | |/ _` | '__|   | |   | |    | |
  / ___ \| | | | (_| | |_| | | (_| | |      | |___| |___ | |
 /_/   \_\_| |_|\__, |\__,_|_|\__,_|_|       \____|_____|___|
                |___/

Angular CLI: 18.2.8
Node: 20.18.0
Package Manager: npm 10.9.0
OS: win32 x64

Angular: undefined
...

Package                      Version
------------------------------------------------------
@angular-devkit/architect    0.1802.8 (cli-only)
@angular-devkit/core         18.2.8 (cli-only)
@angular-devkit/schematics   18.2.8 (cli-only)
@schematics/angular          18.2.8 (cli-only)
```

3. If Angular is not installed, run one of the following commands. Which one you run depends on your operating system.

    Windows: `npm i -g @angular/cli`

    Mac/Linux: `sudo npm i -g @angular/cli`

You may get prompted to share anonymous data. The choice is yours.

4. Once finished, return to step 1 to confirm Angular has installed

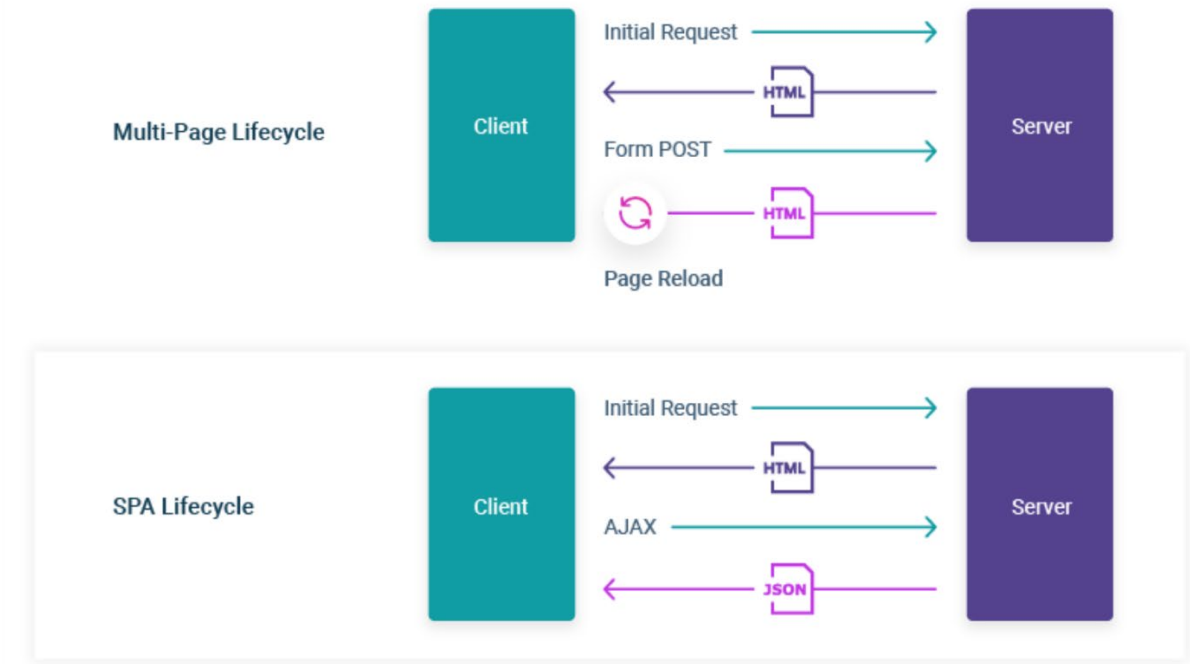
Overview

Angular uses a "single-page application" paradigm to create interactive web applications. In this paradigm, most of the rendering work happens in JavaScript in the browser. This is in contrast to traditional server-rendered web applications, which you may be familiar with (e.g. ASP.NET MVC, Spring MVC).

Single page application (client-side rendering)	Multi page application (server-side rendering)
• Loads only one HTML page. • Updates the content dynamically using JavaScript • No full page reloads after the initial load. • Examples: Angular, React, Vue, Ember	• Loads a new HTML page for each different Page content. • Reloads the entire page when navigating Between pages. • Can feel less responsive and fluid than SPAs. Examples: ASP.NET MVC, Spring MVC, Django Ruby on Rails

Multi page application; Each action must be processed by the server, and an entire new page must be sent back.

Single page application: All HTML & JavaScript is loaded when the user first comes to the site. After that, each action is processed in the browser without reloading the page. If the application needs to interact with a database or additional data, it communicates with the server using APIs.



Creating an Angular Application

a. In GitBash, change to your `classwork/Unit-7` folder:

cd ~/workspace/classwork/Unit-7-Angular

b. Generate a new Angular project: **ng new myFirstAngularApp**

If you get the following questions answer them as indicated (ie. **CSS** and **no**)

? Which stylesheet format would you like to use? CSS

? Do you want to enable Server-Side Rendering (SSR) and Static Site Generation (SSG/Prerendering)? no

c. Change to the Angular project folder: **cd myFirstAngularApp**

d. List the files in the Angular project folder: **ls -a**

e. Run the Angular application: **ng serve**

(or **ng serve -o** *to open in browser immediately)*

Expected output:

```
Application bundle generation complete. [2.137 seconds]
Watch mode enabled. Watching for file changes...
➜ Local: http://localhost:4200/
➜ press h + enter to show help
```

f. Click on the server URL: **http://localhost:4200/** to run the app in your browser. (not necessary if you issued **ng serve -o** command**)**

g. Go to your browser to see the Angular App page.

h. Stop the server, in the session started the app: **ctrl-c** *(it may take a few seconds)*

h. Be sure port used by the server is released: **npx kill-port 4200**

i. Open your Angular Project in VS Code: **code .**

Files and Folders Created for an Angular App *(1 of 2)*

These files and folders are created and populated when creating a new Angular application using the **ng new** command. The **.ts** extension indicates a TypeScript file.

node_modules: Contains the installed dependencies;

- Must exist for **ng serve** command to work
- May be created by **npm install**, if missing
- This file is quite large and is typically not saved to GitHub
- *(should be included in **.gitignore** file)*
- Not included in production deploy

src: Folder containing source code for the application;
Angular components, services, and other code that makes up the application;
Contains the **app** folder

app: Folder containing main Angular application code; Files in this folder:

app.component.css: CSS for your app component

app.component.html: Html for the app component; Template file to do the data binding

app.component.spec.ts: Unit testing file for the app component

app.component.ts: Defines the root component of the application;
Most important typescript file;
Includes the view logic behind the component.

app.config.server.ts: Configure a Node.js server;
Responsible for bootstrapping the Angular application and
Providing it with the necessary configuration.

app.config.ts: Tell Angular how to assemble the application;
Defines the application's root component - entry point for the application;
Defines the application's providers - services used in the application

app.routes.ts: Routing configuration for the application;
Defines all the possible routes in the application; specifies which component should be displayed for each route.

app.module.ts: Configures the application modules;
All the dependencies for the website;
Used to define the needed modules to be imported, the components to be declared and the main component to be bootstrapped
No longer required after Angular 17

assets: Folder containing static assets, such as images, fonts, and CSS files;
Assets are typically referenced in the application's HTML and CSS files.

favicon.ico: Small icon that is displayed in the browser's tab bar;
Typically a 16x16 pixel image, but can be up to 32x32 pixels

index.html: Primary HTML file for the application;
The basic structure of the application;
Loads the Angular application code.

Files and Folders Created for an Angular App *(2 of 2)*

favicon.ico: Small icon that is displayed in the browser's tab bar;
Typically a 16x16 pixel image, but can be up to 32x32 pixels

main.server.ts: Entry point for server-side rendering (SSR);
Responsible for configuring and starting the server-side rendering process

main.ts: Entry point for an Angular application;
Responsible for bootstrapping the application and loading the root module

styles.css: CSS styling to be applied to all pages in an application

angular.json: Configuration options for the Angular CLI;
i.e Configure the build process, the development server, and other aspects of the Angular CLI.

package-lock.json: Lock down the versions of your dependencies so that a project will always use the same versions, regardless of when they were installed

package.json: Information about the project's dependencies;
Used by the Angular CLI to install/manage dependencies, and to run scripts.

README.md: Contains description or application, directions on use and any other information to be conveyed to a user of the application.

server.ts: Used to run an Angular application on a server;
Responsible for starting the Node.js server/configure it to serve the Angular app;
May also include code to handle routing and to provide access to data in a database.

tsconfig.json: Specifies the TypeScript compiler options for the project;
Control how the TypeScript compiler transpiles TypeScript into JavaScript

tsconfig.app.json: Additional TypeScript compiler options for the project;
Extends/Adds to the options in **tsconfig.json**

tsconfig.spec.json: Additional TypeScript compiler options for the project;
Extends/Adds to the options in tsconfig.json

environments: Folder containing environment-specific configuration files;
Used to configure different aspects of the application ie. API endpoint URLs, for different environments *(development, staging, and production)*
Not required for Angular application to run

What is a Component?

Multi page applications are typically composed of pages. Angular applications are composed of **components**. A component is a building block of a webpage. Using an online store as an example, there will be the following components.

- A header component that displays the different pages you can visit.
- A store component that is our main page.
- A cart component that displays all items we bought.
- An item component that displays a single item.

Each component encapsulates its own user interface, data, and logic. Then these components are arranged together to make a complete application.

To Create a component:

1. Navigate to the folder with the Angular Application.
2. Issue the command: `ng g c components/component-name`

   (Short for: `ng generate component components/`*component-name*)

Example:

```
ng g c components/new-component
```

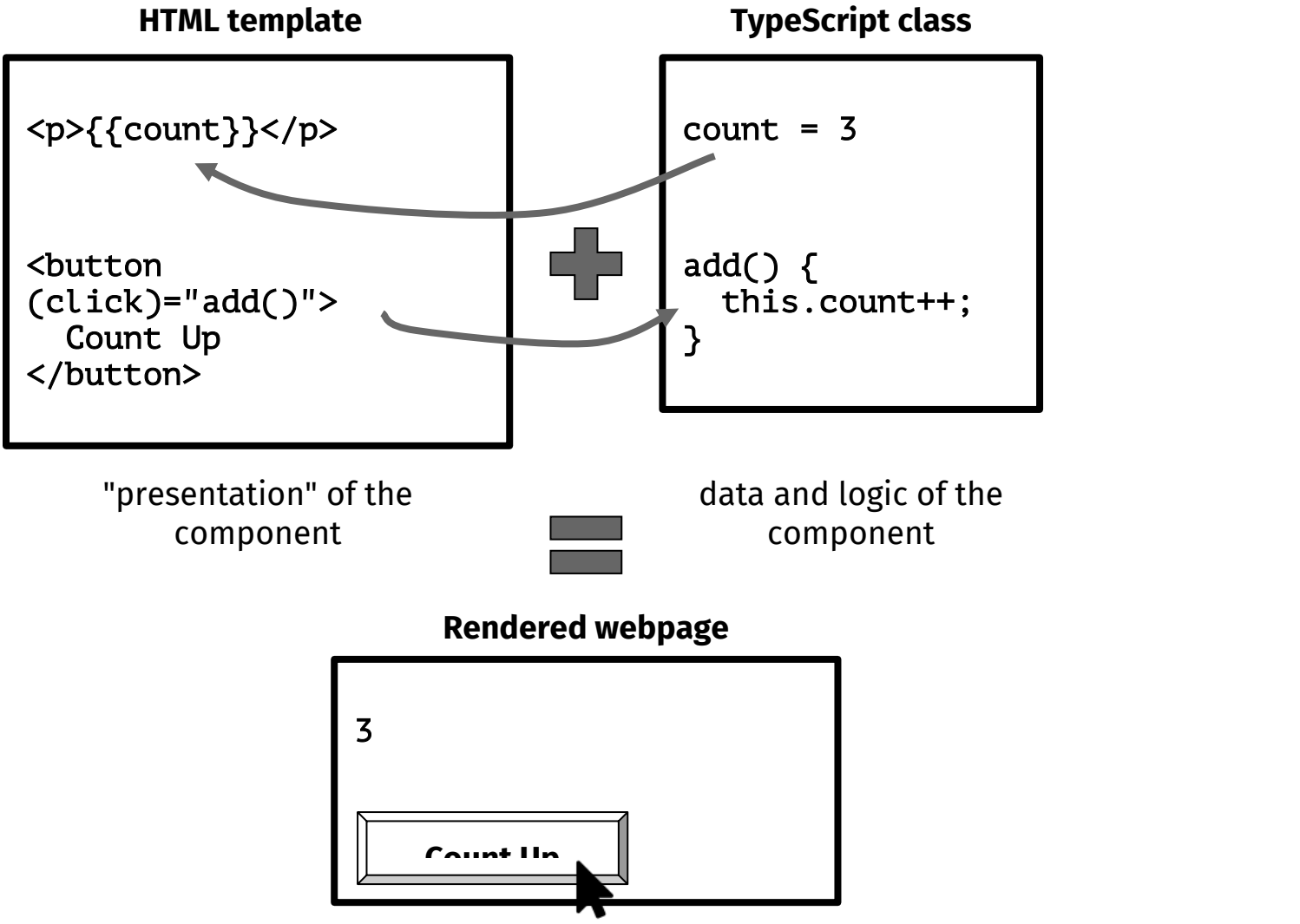
```
CREATE src/app/components/new-component/new-component.component.html (29 bytes)
CREATE src/app/components/new-component/new-component.component.spec.ts (658 bytes)
CREATE src/app/components/new-component/new-component.component.ts (273 bytes)
CREATE src/app/components/new-component/new-component.component.css (0 bytes)
```

Anatomy of a component

Each component has three key parts:

1. A TypeScript class that has properties for data and methods for logic.
2. An HTML template that displays the data from the class and calls its methods when the user interacts with the page.
3. CSS to style the HTML.

Here is an illustration of how the TypeScript class and HTML template work together to produce the rendered component that users interact with. The rest of this lesson summary goes into more details on how this works.



Additional Resources

- https://angular.dev/ - Official Angular website for documentation. Try working through the tutorial!
- Template syntax (Angular Docs)