Using Typescript in your SPFx Webpart

1. To create a new web part project

* Create a new project directory in your favourite location.
* Open command prompt
  + Run as administrator

***md TypescriptWebpart***

1. Go to the project directory

*cd* ***TypescriptWebpart***

1. Create a new ***TypescriptWebpart*** by running the Yeoman SharePoint Generator.

yo @microsoft/sharepoint

**When prompted**:

*Text

Description automatically generated*

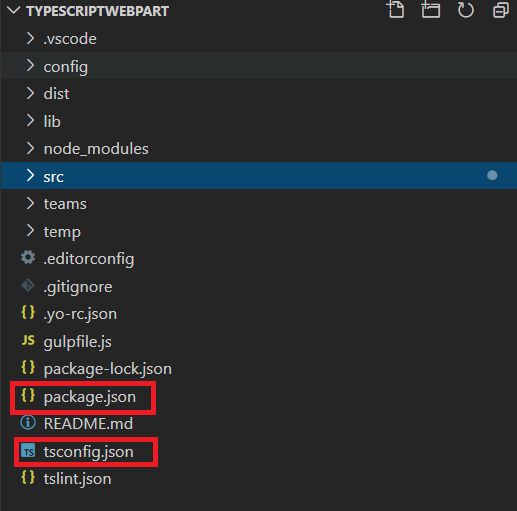
*Text

Description automatically generated*

After created the webpart, open the solution in Visual studio code

code .

1. Now you know the files highlighted below



1. tsconfig.json to manage typescript execution
2. package.json to manage dependencies and starting point
3. Create a Components folder under src i.e src 🡪 Components
4. Then create two files under component folder
   1. IPerson.ts
   2. module1.ts
5. Open IPerson.ts and paste below code

export interface IPerson {

fristname: string;

lastname: string;

}

1. Open module1.ts and paste below code

export let age : number = 20;

export let strval : string = "Data from module";

export class employee {

private empCode: number;

private empName: string;

constructor(name: string, code: number) {

this.empName = name;

this.empCode = code;

}

public displayEmployee() {

console.log ("Employee Code: " + this.empCode + ", Employee Name: " + this.empName );

return("Employee Code: " + this.empCode + ", Employee Name: " + this.empName );

}

}

1. FYI : In module1.ts file I have added one class-employee, one string variable - strval and one number variable - age
2. Save all
3. Open HelloWorldWebPart.ts file
4. Import module

import \* as employeeinfo from'./../../Components/module1';

import {IPerson} from './../../Components/IPerson';

1. Add a constant value to pass interface object

const username = {

fristname: "Jenkins",

lastname: "NS"

};

1. Find **public render(): void {**

add below code next to render():void

let fullname = this.welcomeuser(username);

let empinfo = new employeeinfo.employee('Oliver', 10);

let empinforeturn = empinfo.displayEmployee();

1. Replace this.domelement.innerHtml = `… up to end

</section>`;

    this.domElement.innerHTML = `

    <section class="${styles.myTypescriptexample} ${!!this.context.sdks.microsoftTeams ? styles.teams : ''}">

      <div class="${styles.welcome}">

        <img alt="" src="${this.\_isDarkTheme ? require('./assets/welcome-dark.png') : require('./assets/welcome-light.png')}" class="${styles.welcomeImage}" />

        <h2>Well done, ${escape(this.context.pageContext.user.displayName)}!</h2>

        <div>${this.\_environmentMessage}</div>

        <div>Web part property value: <strong>${escape(this.properties.description)}</strong></div>

      </div>

      <div>

        <h3>Welcome to SPFx Typescript Example!</h3>

        <p>${fullname}</p>

        <p>${emplreturn}</p>

        <p>${emplyInfo.age}</p>

        <p>${emplyInfo.strval}</p>

      </div>

    </section>`;

1. Add a method below the render method – to call **welcomeuser**

protected welcomeuser(user: IPerson): string

{

console.log(`Hey ${user.fristname} ${user.lastname} - Welcome to SPFx training`);

return `Hey ${user.fristname} ${user.lastname} - Welcome to SPFx training`;

}

1. Compile and run the webpart
2. Open the command prompt or Terminal
3. Type **gulp serve**
4. It will open a localhost workbench
5. And display the output like below

Graphical user interface, text, application

Description automatically generated