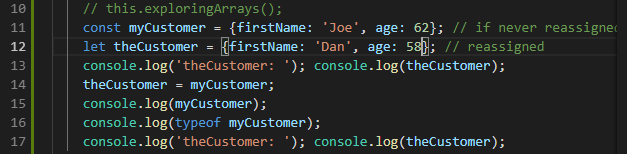
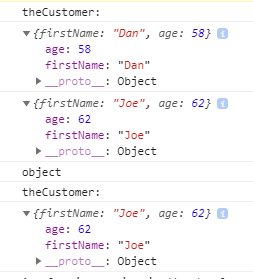
**Javascript Objects**

Use let for reassignments

Use const for never reassigned

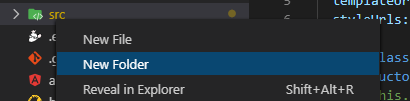


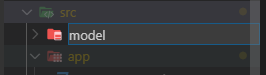


The type is Object - a javascript object

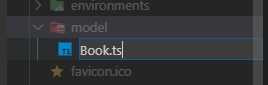
**Creating Classes:**

Create a model folder for the class Book.ts

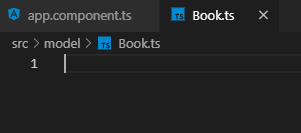




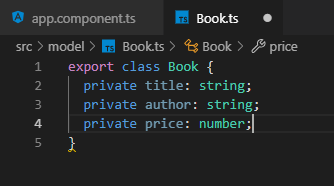




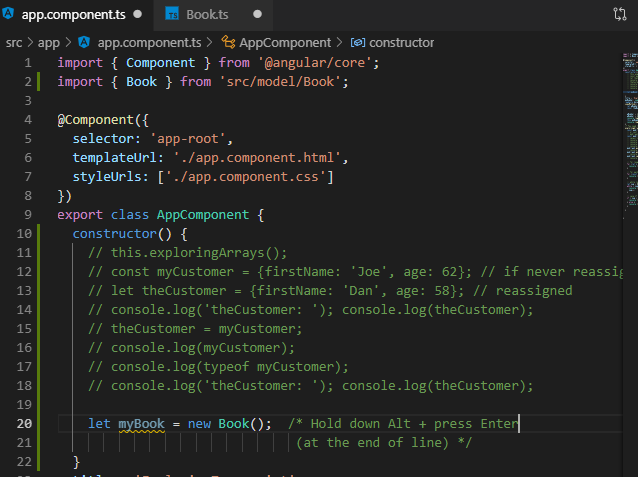
Typescript file for Class Book:



Now, create the public class Book

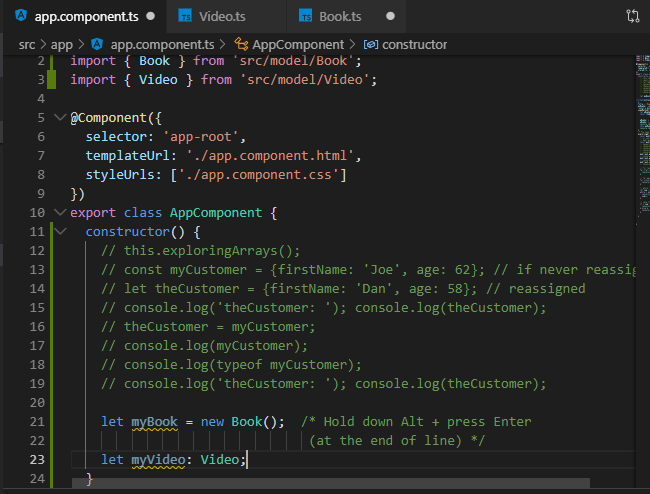


Now import the class into the app.component.ts file



You can have multiple public classes in the same file but that screws up the naming.

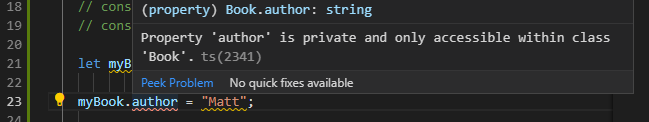
Video class with import successfully from the Book.ts file, but do not do this.



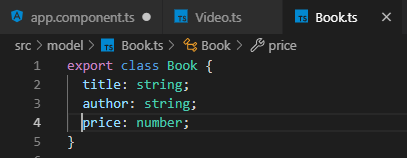
**Making properties of the class accessible:**

All you need to do is to remove the private modifier which makes the properties public:

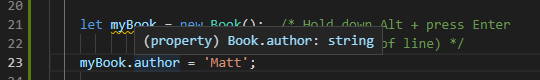
Before removal:



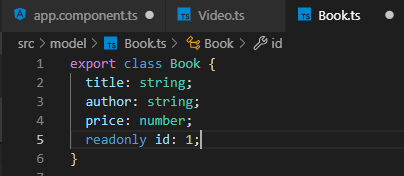
**After removal:**



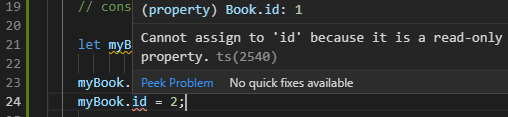
**Accessible:**



**You can modify access with readonly:**



**Cannot set the id**

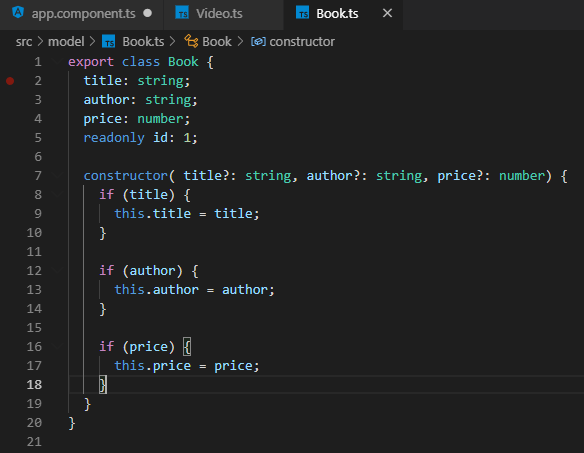


**Creating Constructors and methods:**

**You can only have signal constructors per class,**

**but you can overload a construction with optional parameters (use ?) to get the**

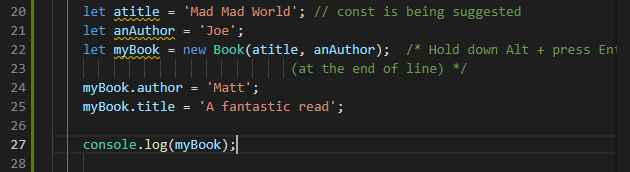
**same effect.**

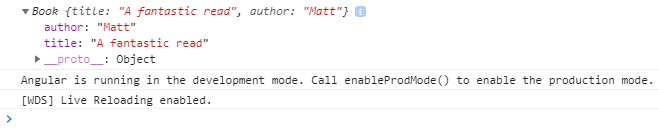


readonly id : 1; should be **readonly id = 1;**

**This class is both parameter-less since all optional and with parameters:**

**Using the constructor:**





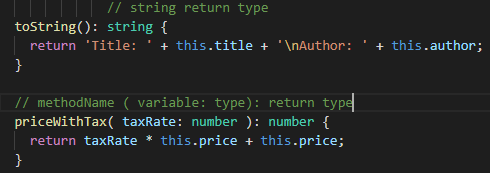
**Let us implement a toString()**

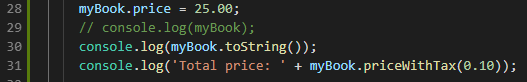






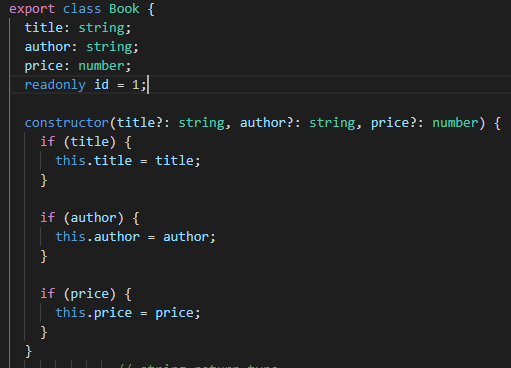
**Methods**

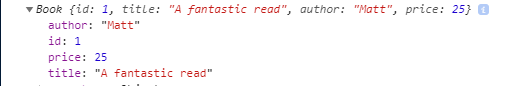




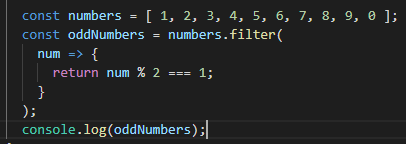


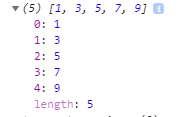
**Getting the id output:**





**Lambda functions or in typescript called arrow functions ( => )**





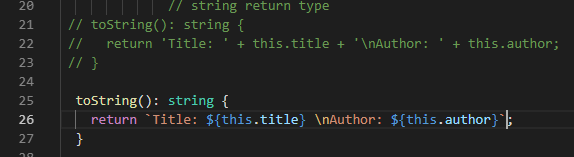
**Shorter syntax**





**String Templates**

**Typescript string embedding technique using backticks instead quotes and ${}**

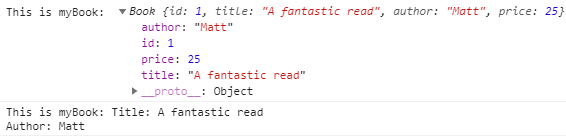




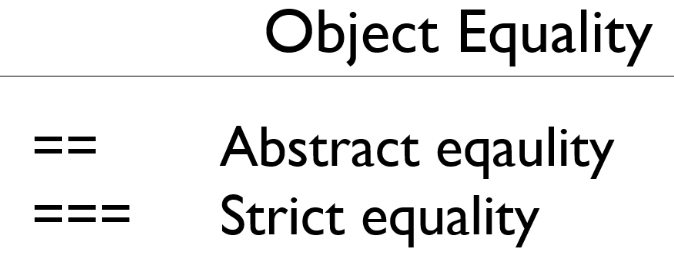
**Debugging Classes**

**Use the console.log( with a comma )**





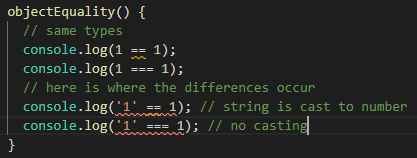
**Object Equality**



**= assign**

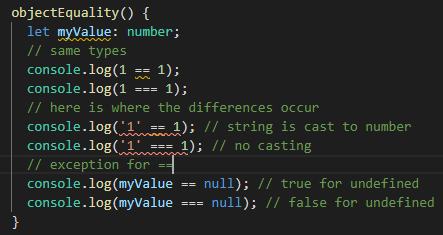
**== cast or conversion attempted, they are equal**

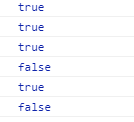
**=== values are equal**





**Use == for the null check cause undefined is not the same as null**





**!=**

**!== use this one for values**