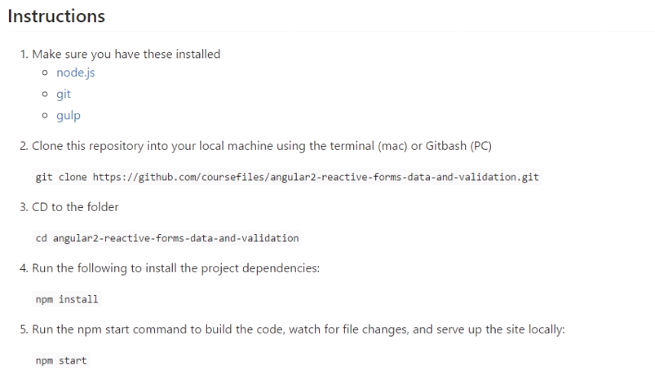
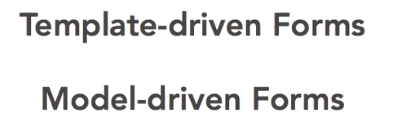
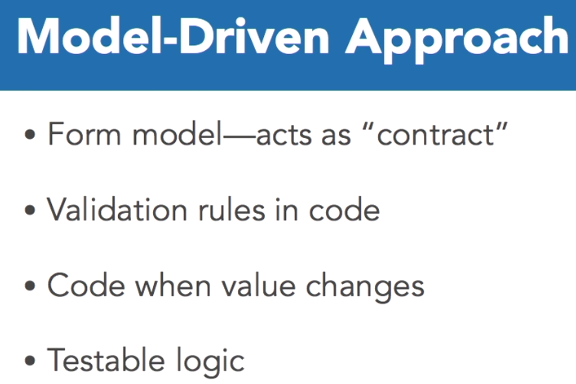
**Steps to run an angular project:**



TypeScript code is based on the ECMAScript 2015 spec, including things like classes, and the Let keyword.

**Angular Form:**





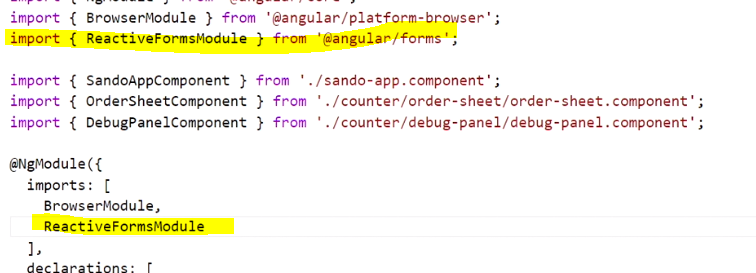
The model-drive approach has several strong benefits.

1. Ability to create a contract that you're form UI needs to adhere to.
2. Ability to setup validation rules.
3. Ability to subscribe and react to field value changes.
4. Opportunity to be able to unit test the logic of form.

To use the model-driven approach to forms, you need to add the ReactiveFormsModule to the list of imports for your Angular module.

The ReactiveFormsModule contains all the form directives and constructs for working with forms.

**AppModule.ts**

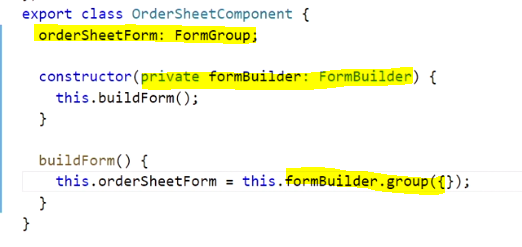


1. Angular provides a class named **FormGroup** that is the **foundation of a form model**.
2. Representing a form in code begins with an instance of a FormGroup.
3. First you need to import the FormGroup class from Angular.
4. FormGroup is found in the angular forms scope package.
5. FormGroups are used for the base of the form model representation, but they are also used to logically group form controls together. A FormGroup will aggregate the values of the form controls in it, so its status will depend on the status of its children.

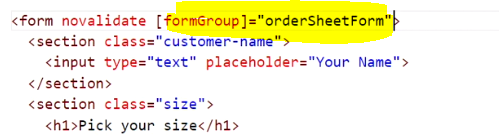
There are **two** ways we can create a FormGroup.

1. We can instantiate a new instance of it using the FormGroup class.
2. We can use a class that Angular exports named FormBuilder that has some methods on it to create FormGroups, and even FormControls.

To use the FormBuilder, you can add it to the constructor parameters, to have Angular's dependency injection provided to the class.



**Wiring of with the html file:**



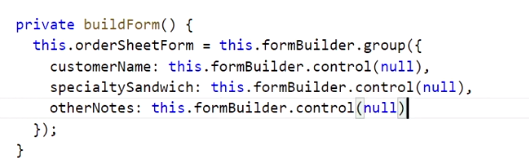
To add form fields to the form model, we can use a class that Angular provides, named Form Control.

Representing form elements in code involves creating them as form control objects.

A form control can be thought of as a baseline building block for a form.

Form control is what we use to represent form elements that will collect data from users.

Ex:



**Form Array:**

The form array is another fundamental building block that Angular provides for working with forms.

Like a form group, a form array provides a logical grouping and aggregates the values of each nested form control in it, allowing you to track the state of the group. But one thing unique to a form array is that it's designed to support dynamically adding and removing form controls to it.

Wiring up to a select element is straightforward. All you need to do is put the form control name attribute on it and set it equal to the property name for that form control.

Fundamental building blocks for forms:

1. formControl

2. formGroup

3. formArray