Cheatography

Angular 2 Forms Cheat Sheet

by Nathan (Nathane2005) via cheatography.com/28056/cs/8477/

Introduction

A form creates a cohesive, effective, and compelling data entry experience. An Angular form coordinates a set of data-bound user controls, tracks changes, validates input, and presents errors.

Form

```
<form>
   .... tags that include all input elements
</f orm>
```

All forms are placed within the HTML form tags

Standard Input Types

Text Input	<input type="text">
Email Input	<input type="email">
Password Input	<input type="password">
Dropdown Selection	<select> <option value="volvo">Volvo</option> <option value="saab">Saab</option> <option value="opel">Opel</option> <option value="audi">Audi</option> </select>
Multi Selection	<select multiple> <option value="volvo">Volvo</option> <option value="saab">Saab</option> <option value="opel">Opel</option> <option value="audi">Audi</option> </select>
Checkbox	<input type="checkbox">
Radio Control	<input type="radio">
Numeric Input	<input type="number">
Date	<input type="date">
Multiline Input	<textarea rows="4" cols="50"></textarea>

Angular 2 Form - Elements

Angular 2 Form - Elements (cont)

FormArray	Tracks the value and validity state of an array of FormControl instances. A FormArray aggregates the values of each child FormControl into an array
FormBuilder	Creates an AbstractControl from a user-specified configuration. It is essentially syntactic sugar that shortens the new FormGroup(), new FormControl(), and new FormArray() boilerplate that can build up in larger form

Requires use of FormModule

Reactive Form Names

formGroupName	Used to reference a group of elements
formControlName	Similar to ngModel reference to a name but simpler from a naming convention perspective
formArrayName	Syncs a nested FormArray to a DOM element.

Requires the use of the ReactiveFormsModule Module

Handling Submission Event

```
<form (ngSubmit)="onSubmit()">
...
</f orm>
```

Standard Validation

Mandatory	Validators.required
Minimum Length	Validator.minLength(size)
Maximum Length	Validators.maxLength(size)
Pattern Match	Validators.pattern("regEx")

Custom Validators

Displaying Validator Failures

```
<label for="name">Name</label>
                <input type="t -
ext " class= " for m-c ont -
rol " id=" nam e"
                                -
req uired
                                -
[(n gMo del )]= " mod el.n am e"
name="n ame "
                                -
#na me= " ngM ode l" >
                <div [hidde -
n]= " nam e.valid || name.p -
ris tin e"
                            cla -
ss= " alert alert- dan ger ">
                    Name is
required
                </d iv>
```

Workflow

Steps to creating a reactive form:

1. Create the Domain Model
2. Create the Controller with references to View
3. Create the View
4. Add Validations
5. Add Submit Validation Control
6. Add Dynamic Behaviors

Model

```
export interface {ModelName} {
      item(? : optional) :
string | number | date | boolean
| class | interface ([] :
array);
}
```

Controller

FormGroup	A FormGroup aggregates the values of each child FormCo-ntrol into one object, with each control name as the key
FormCo-ntrol	Tracks the value and validation status of an individual form control. It is one of the three fundamental building blocks of Angular forms

```
function {name}(control :
FormControl) : {[s: string] :
boolean} {

.... function body....
pass return a null
fail return an object of type
{key : true}
}
```

```
let style =
require('./someStyle.css');
let template = requir e("./ so -
meT emp lat e.h tml ");
@Compo nent({
    sty les :[s tyle],
    tem plate: template
});
export class {Some}Form
implements OnInit{

    myForm: FormGroup;
    con str uct or( private fb :
FormBu ilder) {};
    ngO nInit() {
        //C ons truct the form
data type

      thi s.m yForm: this.f -
b.g roup({
              'co ntr olName' :
this.f b.c ont rol (...),
              'co ntr olA rray'
: this.f b.a rra y([...]),
              'co ntr olG roup'
: this.f b.g rou p({})
   });
   }

  onS ubmit() {
      myF orm.value; //returns
the form values
```

By **Nathan** (Nathane2005)

cheatography.com/nathane2005/

Published 9th October, 2016.
Last updated 9th October, 2016.

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com

Controller (cont)

```
>   myModel = <MyModel>myForm.va-
lue;//Cast to object
 }
}
```

Typical additions include:

1. Http Service Submission (delegate normally injected)
2. Pipes for Display customization
3. Model based Validators

View

```
<form [formGroup]='myForm'
(ngSubmit)='onSubmit()'>
    <input formCo ntr olN ame -
=''>
    <div formGr oup Nam e=' '>
            <input formCo ntr -
olN ame =''>
    </d iv>
    <div formAr ray Nam e=' '>
            <input
formControlName='{{index}}'
                *ng For ='let
item of items; index = index'>
    </d iv>
</f orm>
```

Useful Blocks

```
-- Get Form Items
JSON.s tri ngi fy( myF -
orm.value)
```

Useful Links

Angular Forms
TypeScript Basic Types
HTML Inputs

Sponsored by **CrosswordCheats.com**
Learn to solve cryptic crosswords!
http://crosswordcheats.com