CaseStudyNotes-Angular-ShoppingCart:

<https://medium.com/codex/how-to-share-data-between-components-in-angular-a-shopping-cart-example-b86ce8254965>

<https://www.baeldung.com/spring-boot-angular-web>

<https://blog.angular-university.io/introduction-to-angular-2-forms-template-driven-vs-model-driven/>

Everything about the forms:

**Enabling Template Driven Forms is close to AngularJS**

**ngModel; ngForm. ngSubmit**

## Angular Reactive Forms: more preferred form for Angular?

import {ReactiveFormsModule} from "@angular/forms";

HTML reactive form:

<section class="sample-app-content">

<h1>Reactive Form Example:</h1>

<form [formGroup]="form" (ngSubmit)="onSubmit()">

<p>

<label>First Name:</label>

<input type="text" formControlName="firstName">

</p>

<p>

<label>Password:</label>

<input type="password" formControlName="password">

</p>

<p>

<button type="submit" [disabled]="!form.valid">Submit</button>

</p>

</form>

</section>

1. there is a formGroup directive applied to the whole form, binding it to a component variable named form.
2. the required validator attribute is not applied to the form controls. This means the validation logic must be somewhere in the component class, where it can be more easilly unit tested.

**import { FormGroup, FormControl, Validators, FormBuilder }**

**from '@angular/forms';**

**@Component({**

**selector: "reactive-form",**

**templateUrl: 'reactive-form.html'**

**})**

**export class ReactiveFormExample {**

**form = new FormGroup({**

**"firstName": new FormControl("", Validators.required),**

**"password": new FormControl("", Validators.required),**

**});**

**onSubmitModelBased() {**

**console.log("reactive form submitted");**

**console.log(this.form);**

**}**

**}**

**FormGroup will keep track of the global form value and the validity state.**

**The controls themselves can be instantiated individually using the FormControl**

**Constructor.**

The end result is a programmatic definition of our form model with all of its controls and validity rules, that is created programmatically at the level of the component class, and not the template.

### The FormBuilder API: the built-in FormBuilder service

import { FormGroup, FormControl, Validators, FormBuilder }

from '@angular/forms';

@Component({

selector: "reactive-form",

templateUrl: 'reactive-form.html'

})

export class ReactiveFormExample {

form = fb.group({

"firstName": ["", Validators.required],

"password":["", Validators.required]

});

constructor(fb: FormBuilder) {

}

onSubmitModelBased() {

console.log("reactive form submitted");

console.log(this.form);

}

}

Reactive Forms are a much better choice!

Case 1:

Main Points: share the data between components using the observables. Mainly BehaviorSubjects. Also add backend webservice support.

1. Create a new angular project.
2. Create all the components.
3. Create routing module.
4. Create services.