

Checkout Form - Layout

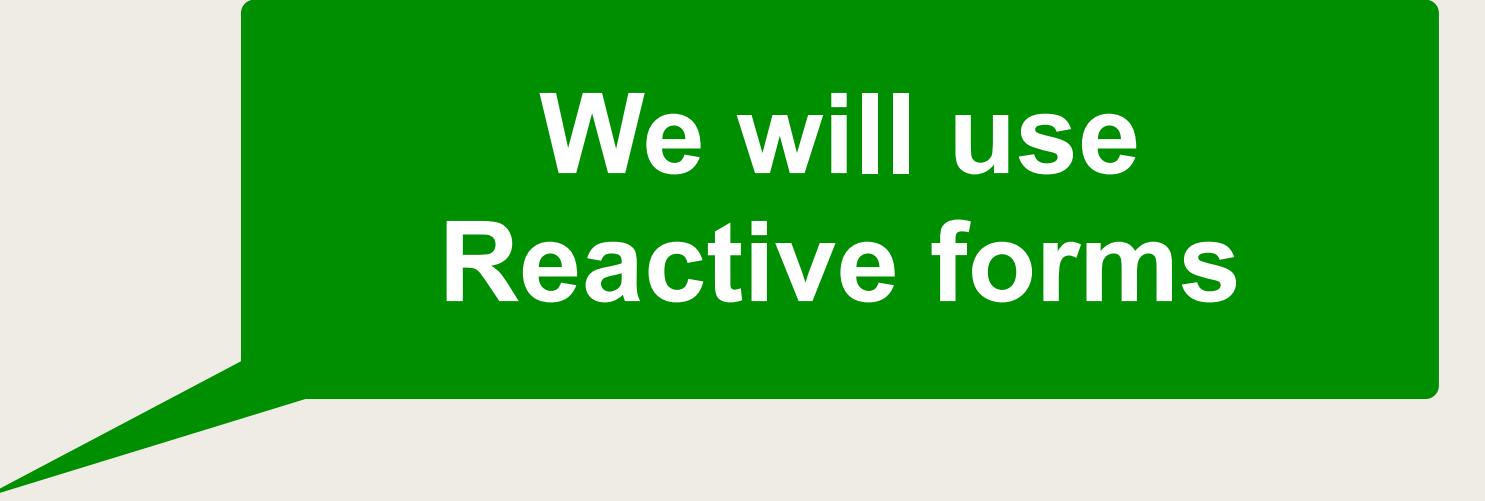


Overview of Angular Forms

- Can easily build forms in Angular
- Supports form data-binding, validation and processing
- Angular provides two types of forms
 - Reactive forms
 - Template-driven forms

Forms

- **Reactive forms**
 - Leverages programmatic API for form building
 - Scalable solution that is designed for large, complex forms
 - Forms can be easily reused and tested
- **Template-driven forms**
 - Targeted for small, simple forms
 - Not a scalable solution for large, complex forms



We will use
Reactive forms

Comparison

- For comparison of Reactive forms and Template-driven forms

<https://angular.io/guide/forms-overview>

Key Components

| Name | Description |
|----------------------|--|
| FormControl | Individual control that tracks the value and validation status |
| FormGroup | A collection of controls. Can create nested groups. |
| <i>others</i> | ... |

Our Checkout Form

Focus on
form construction
and layout

The image displays three screenshots of a checkout form interface for 'luvshop'. The top screenshot shows the main header with a search bar and navigation links for Books, Coffee Mugs, Mouse Pads, and Luggage Tags. Below the header are two sections: 'Customer' (First Name, Last Name, Email) and 'Shipping Address' (Country, Street, City, State, Zip Code). The middle screenshot shows a modal or dropdown for 'Billing Address' (Country, Street, City, State, Zip Code) and 'Credit Card' (Card Type, Name on Card, Card Number, Security Code, Exp Month, Exp Year). The bottom screenshot shows a summary section for 'Review Your Order' with fields for Total Quantity (0), Shipping (FREE), and Total Price (\$0.00), along with a 'Purchase' button.

luvshop

Search for products ...

Books

Coffee Mugs

Mouse Pads

Luggage Tags

Customer

First Name

Last Name

Email

Shipping Address

Country

Street

City

State

Zip Code

Billing Address

Country

Street

City

State

Zip Code

Credit Card

Card Type

Name on Card

Card Number

Security Code

Exp Month

Exp Year

luvshop

Search for products ...

Books

Coffee Mugs

Mouse Pads

Luggage Tags

luvshop

Search for products ...

Card

Card Number

Security Code

Exp Month

Month

Exp Year

Year

Review Your Order

Total Quantity: 0

Shipping: FREE

Total Price: \$0.00

Purchase

Development Process

Step-By-Step

1. Generate our checkout component
2. Add a new route for checkout component
3. Create a new checkout button and link to checkout component
4. Add support for reactive forms
5. Define form in component .ts file
6. Layout form controls in HTML template
7. Add event handler for form submission

Step 1: Generate our checkout component

```
> ng generate component components/checkout
```

Step 2: Add a new route for checkout component

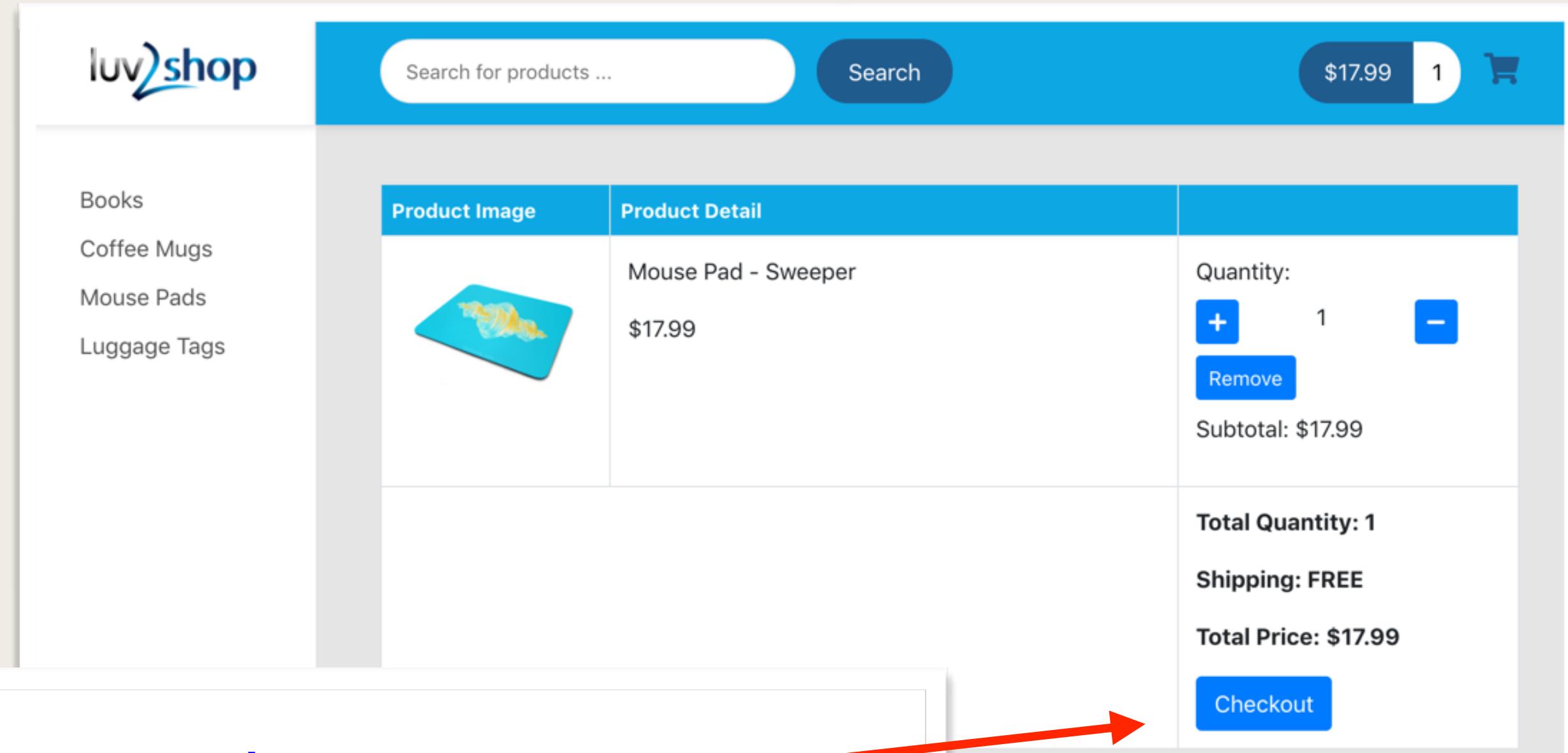
File: app.module.ts

```
const routes: Routes = [
  {path: 'checkout', component: CheckoutComponent},
  ...
];
```

Step 3: Create new checkout button and link to checkout component

File: cart-details.component.html

```
...  
<a routerLink='/checkout' class="btn btn-primary">Checkout</a>  
...
```



The screenshot shows a shopping cart interface for 'luvshop'. At the top, there's a navigation bar with the logo 'luvshop', a search bar, and a cart icon showing 1 item for \$17.99. The main area displays a single item in the cart: 'Mouse Pad - Sweeper' priced at \$17.99. Below the cart summary, there are details: 'Total Quantity: 1', 'Shipping: FREE', and 'Total Price: \$17.99'. A prominent blue 'Checkout' button is located at the bottom right of the cart summary. A red arrow points from the highlighted code in the 'cart-details.component.html' file to this 'Checkout' button.

Step 4: Add support for reactive forms

File: app.module.ts

```
...
imports: [
  ...
  ReactiveFormsModule
],
...
]
```

Step 5: Define form in component .ts file

The screenshot shows a web application interface for 'luvshop'. At the top, there is a navigation bar with a logo, a search bar labeled 'Search for products ...', and a 'Search' button. On the left side, there is a sidebar with links to 'Books', 'Coffee Mugs', 'Mouse Pads', and 'Luggage Tags'. The main content area contains two sections: 'Customer' and 'Shipping Address'. The 'Customer' section has three input fields: 'First Name', 'Last Name', and 'Email'. The 'Shipping Address' section has five input fields: 'Country', 'Street', 'City', 'State', and 'Zip Code'. All input fields are represented by white rounded rectangular boxes.

Step 5: Define form in component .ts file

File: checkout.component.ts

```
export class CheckoutComponent implements OnInit {  
  
  checkoutFormGroup: FormGroup;  
  
  constructor(private formBuilder: FormBuilder) { }  
  
  ngOnInit(): void {  
  
    this.checkoutFormGroup = this.formBuilder.group({  
      customer: this.formBuilder.group({  
        firstName: ['',  
        lastName: ['',  
        email: ['']  
      })  
    });  
  }  
  
  onSubmit() {  
    console.log("Handling the submit button");  
    console.log(this.checkoutFormGroup.get('customer').value);  
  }  
}
```

Inject the form builder

Declare our form group

Build the form

Method to call when
submit button is clicked

Step 6: Layout form controls in HTML template

Name of property from component class

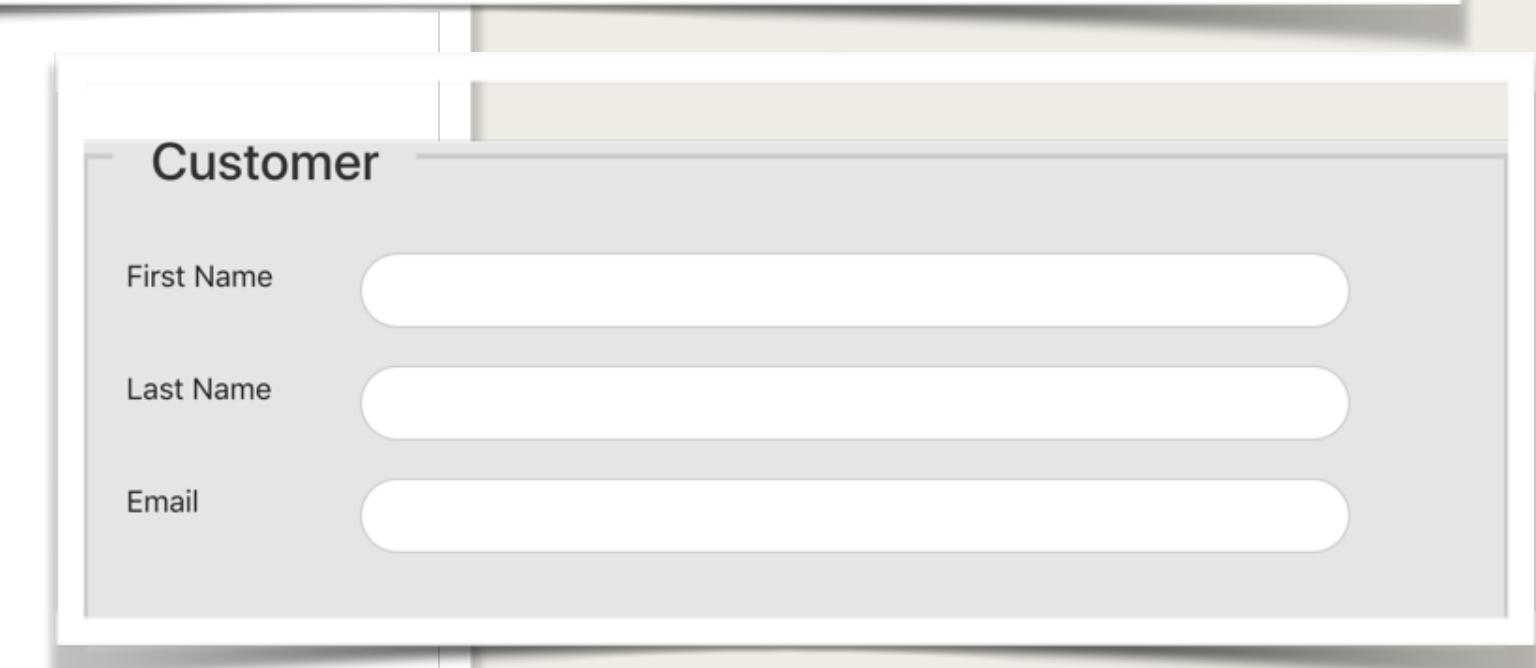
File: checkout.component.html

```
<form [formGroup]="checkoutFormGroup">

    <!-- customer form group -->
    <div formGroupName="customer" class="form-area">
        <h3>Customer</h3>

        <div class="row">
            <div class="col-md-2"> <label>First Name</label></div>
            <input formControlName="firstName" type="text">
        </div>
        ...
    </div>
    ...
</form>
```

```
export class CheckoutComponent implements OnInit {
    checkoutFormGroup: FormGroup;
```



Repeat process for rows:
Last Name, Email

Step 7: Add event handler for form submission

File: checkout.component.html

```
<form [formGroup]="checkoutFormGroup" (ngSubmit)="onSubmit()">  
  
    <!-- customer form group -->  
    <div formGroupName="customer" class="form-area">  
        <h3>Customer</h3>  
  
        <div class="row">  
            <div class="col-md-2"> <label>First Name</label></div>  
            <input formControlName="firstName" type="text">  
        </div>  
    </div>  
  
    <div class="text-center">  
        <button type="submit" class="btn btn-info">Purchase</button>  
    </div>  
...
```

Our submit button

Method to call when
submit button is clicked

```
onSubmit() {  
    console.log("Handling the submit button");  
    console.log(this.checkoutFormGroup.get('customer').value);  
}
```

Read the form data

Step 7: Add event handler for form submission

```
onSubmit() {  
    console.log("Handling the submit button");  
    console.log(this.checkoutFormGroup.get('customer').value);  
}  
->     console.log("Email is " + this.checkoutFormGroup.get('customer').value.email);
```

Read the form data

Output

```
Handling the submit button  
▼ {firstName: "John", lastName: "Doe", email: "john.doe@luv2code.com"} ⓘ  
  email: "john.doe@luv2code.com"  
  firstName: "John"  
  lastName: "Doe"  
▶ __proto__: Object
```

Our Checkout Form

Repeat the process for other sections of the form

The image displays three screenshots of a checkout form for the website "luvshop".

- Customer Section:** Contains fields for First Name, Last Name, and Email.
- Billing Address Section:** Contains fields for Country, Street, City, State, and Zip Code.
- Credit Card Section:** Contains fields for Card Type, Name on Card, Card Number, Security Code, Exp Month, and Exp Year.
- Review Your Order Section:** Displays summary information: Total Quantity: 0, Shipping: FREE, and Total Price: \$0.00. A "Purchase" button is located at the bottom right of this section.

On the left side of the main form, there is a sidebar with links to "Books", "Coffee Mugs", "Mouse Pads", and "Luggage Tags".