Building a Reactive Form

Deborah Kurata

CONSULTANT | SPEAKER | AUTHOR | MVP | GDE

@deborahkurata | blogs.msmvps.com/deborahk/


GIVE
ME
MORE

Module Overview

The Component Class

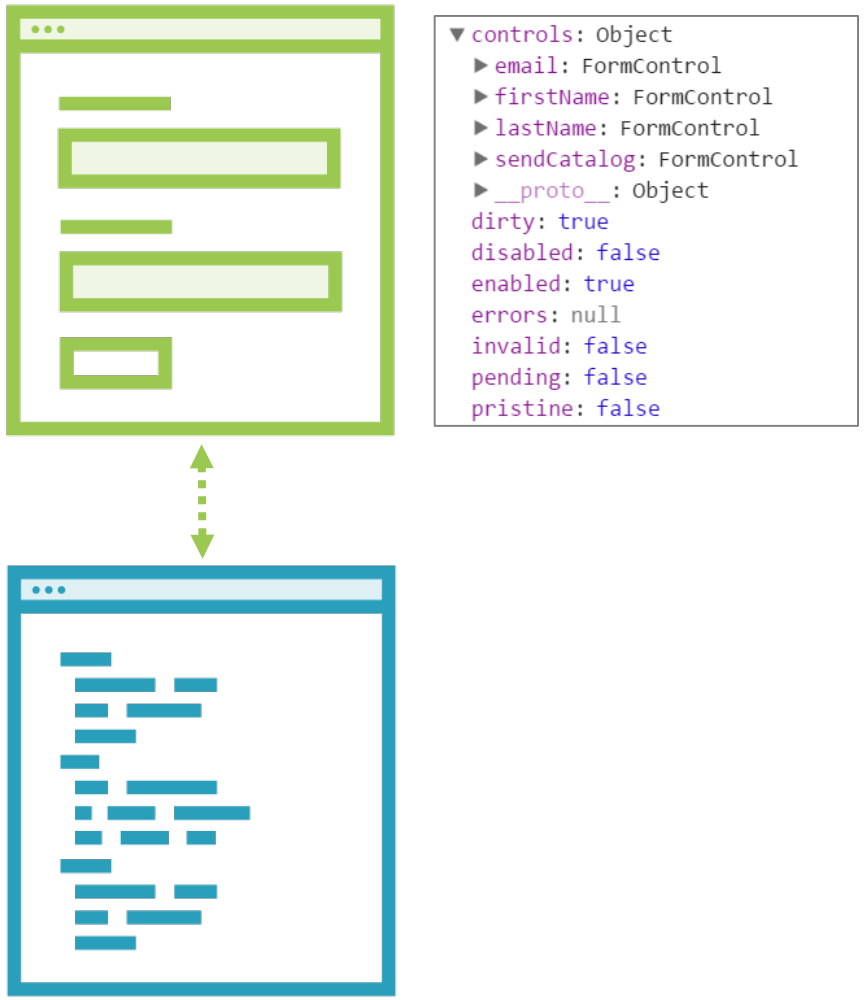
The Angular Module

The Template

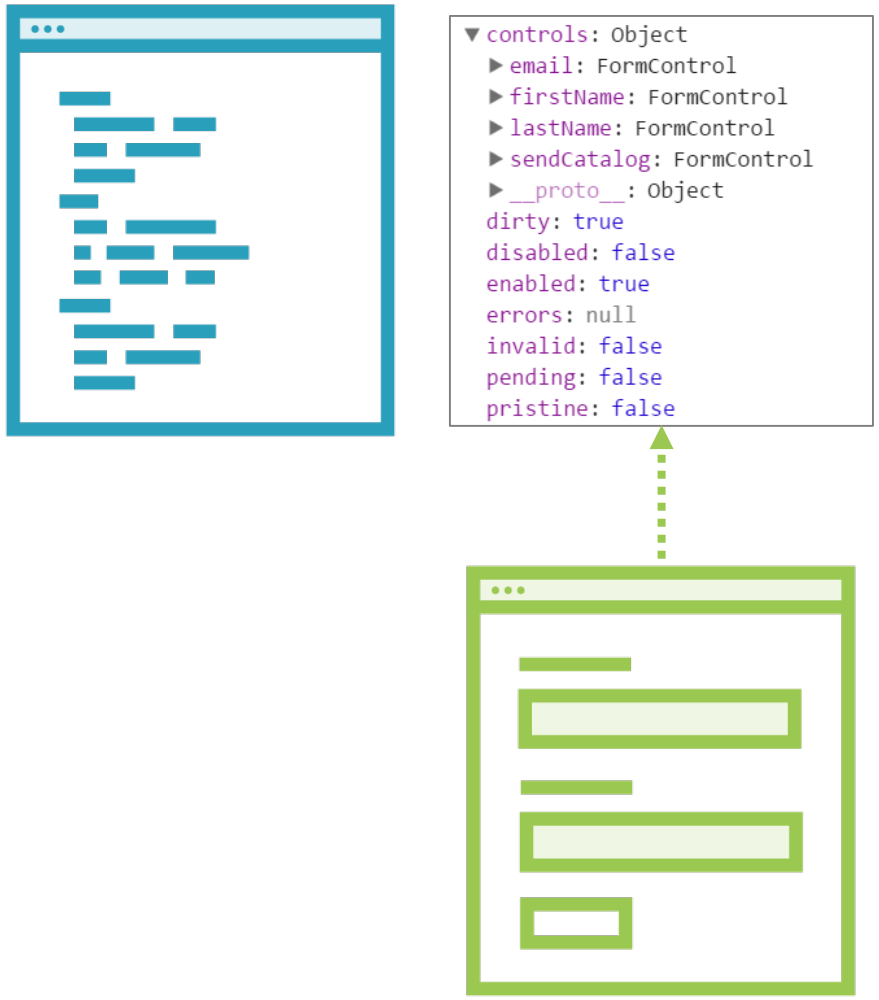
Using `setValue` and `patchValue`

Simplifying with FormBuilder

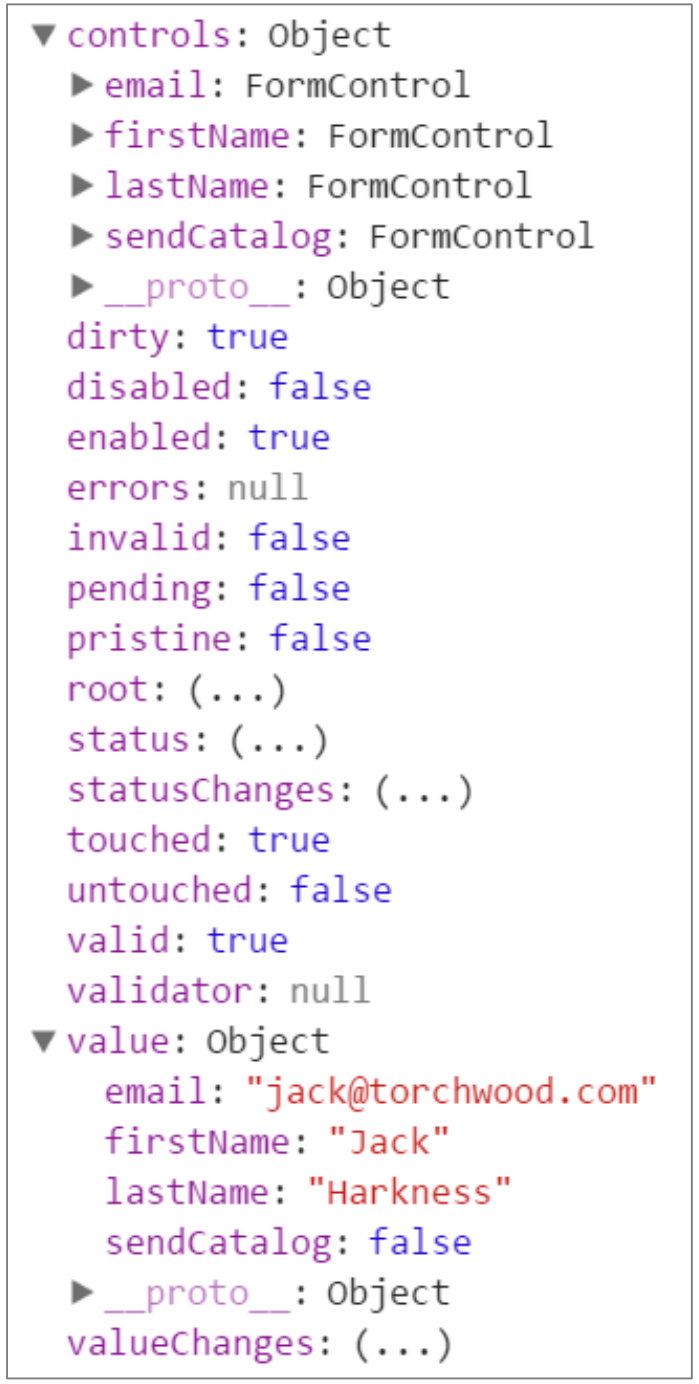
Template-driven vs. Reactive Forms



```
▼ controls: Object
  ▶ email: FormControl
  ▶ firstName: FormControl
  ▶ lastName: FormControl
  ▶ sendCatalog: FormControl
  ▶ __proto__: Object
  dirty: true
  disabled: false
  enabled: true
  errors: null
  invalid: false
  pending: false
  pristine: false
```



```
▼ controls: Object
  ▶ email: FormControl
  ▶ firstName: FormControl
  ▶ lastName: FormControl
  ▶ sendCatalog: FormControl
  ▶ __proto__: Object
  dirty: true
  disabled: false
  enabled: true
  errors: null
  invalid: false
  pending: false
  pristine: false
```



Form Model

- Root FormGroup
- FormControl for each input element
- Nested FormGroups as desired
- FormArrays

Creating a FormGroup

customer.component.ts

```
...
import { FormGroup } from '@angular/forms';


...
export class CustomerComponent implements OnInit {
  customerForm: FormGroup;
  customer: Customer = new Customer();

  ngOnInit(): void {
    this.customerForm = new FormGroup({ });
  }
}
```

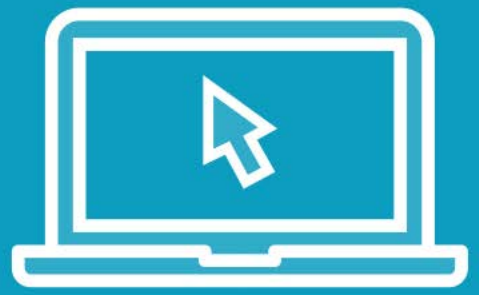
Creating FormControls

customer.component.ts

```
...
import { FormGroup, FormControl } from '@angular/forms';


...
export class CustomerComponent implements OnInit {
  ...
  ngOnInit(): void {
    this.customerForm = new FormGroup({
        firstName: new FormControl(),
        lastName: new FormControl(),
        email: new FormControl(),
        sendCatalog: new FormControl(true)});
  }
}
```

Demo



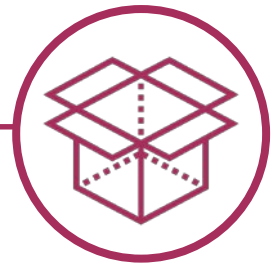
Building the Form Model

AppComponent

Customer-Component

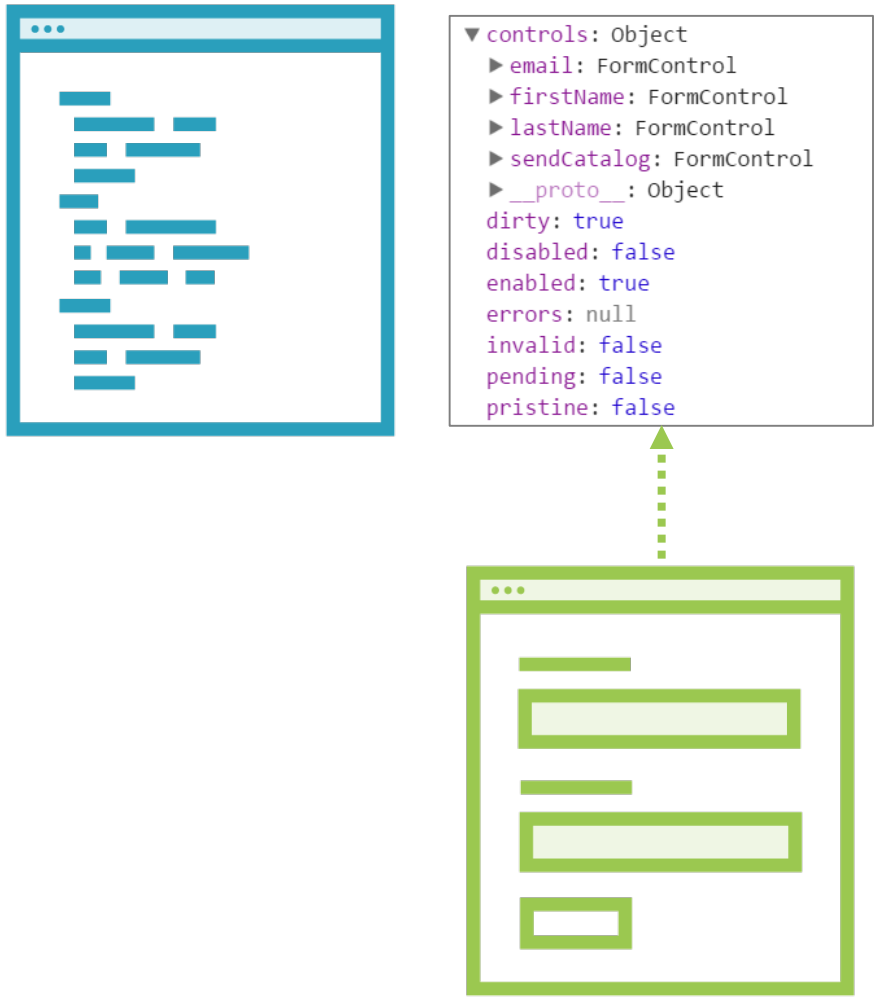
BrowserModule

Reactive - FormsModule



Angular Module

Binding to the Form Model



Reactive Forms Directives

Reactive Forms

- formGroup
- formControl
- formControlName
- formGroupName
- formArrayName

formGroup

customer.component.html

```
<form (ngSubmit)="save()" [formGroup]="customerForm">
    ...
</form>
```

formControlName

customer.component.html

```
<form (ngSubmit)="save()" [formGroup]="customerForm">
  <fieldset>
    <div ... >
      <label for="firstNameId">First Name</label>
      <input id="firstNameId" type="text"
             placeholder="First Name (required)"
             formControlName="firstName" />
      <span ... >
         ...
      </span>
    </div>
    ...
 </fieldset>
</form>
```

Accessing the Form Model Properties

```
customerForm.controls.firstName.valid

customerForm.get('firstName').valid
```

```
firstName = new FormControl();

ngOnInit(): void {
  this.customerForm = new FormGroup({
      firstName: this.firstName,
      ...
  });
}
```

```
firstName.valid
```

Using `setValue` and `patchValue`

```
this.customerForm.setValue({
    firstName: 'Jack',
    lastName: 'Harkness',
    email: 'jack@torchwood.com'
});
```

```
this.customerForm.patchValue({
    firstName: 'Jack',
    lastName: 'Harkness'
});
```

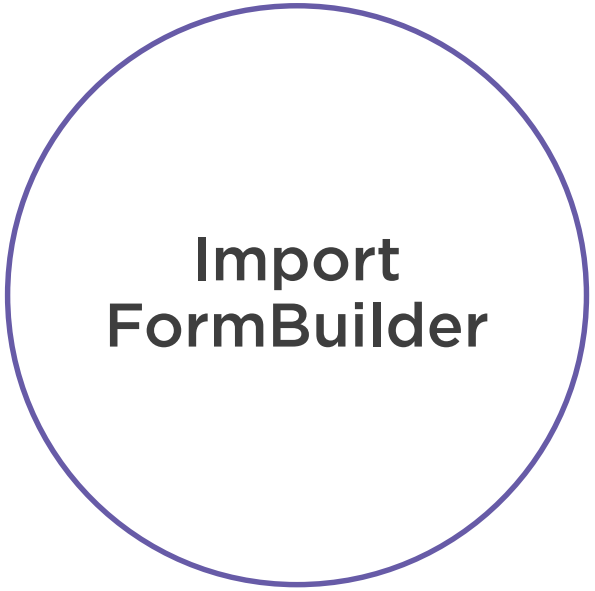
FormBuilder

Creates a form model from a configuration

Shortens boilerplate code

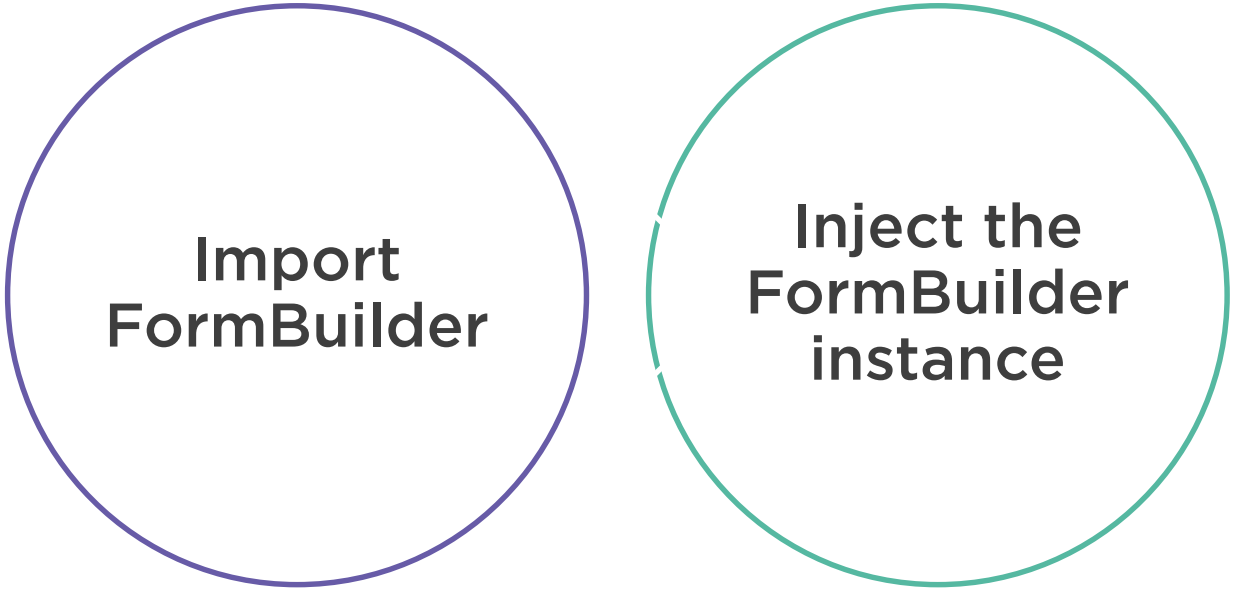
Provided as a service

FormBuilder Steps



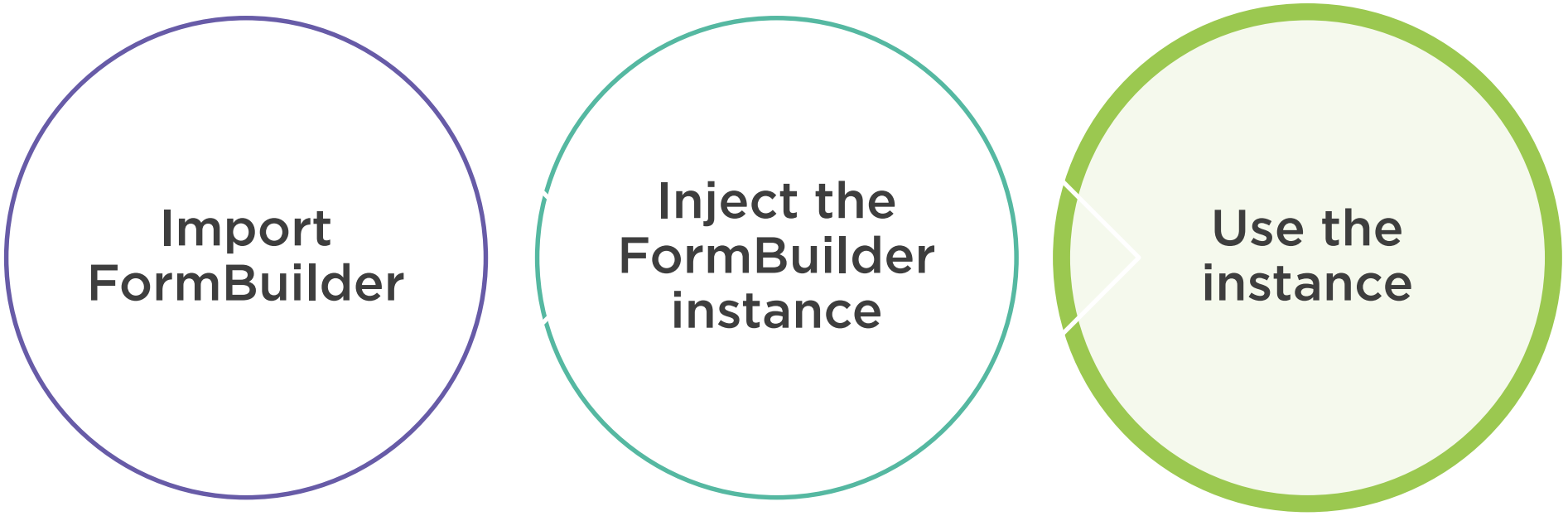

```
import { FormBuilder } from '@angular/forms';
```

FormBuilder Steps




```
constructor(private fb: FormBuilder) { }
```

FormBuilder Steps




```
this.customerForm = this.fb.group({
        firstName: null,
        lastName: null,
        email: null,
        sendCatalog: true
});
```

FormBuilder's FormControl Syntax

```
this.customerForm = this.fb.group({
            firstName: '',
            sendCatalog: true
});
```

```
this.customerForm = this.fb.group({
            firstName: {value: 'n/a', disabled: true},
            sendCatalog: {value: true, disabled: false}
  });
```

```
this.customerForm = this.fb.group({
            firstName: [''],
            sendCatalog: [{value: true, disabled: false}]
  });
```

Checklist: Component Class

Create a property for the root FormGroup

Create the FormGroup instance

Pass in each FormControl instance

```
ngOnInit(): void {
  this.customerForm = new FormGroup({
        firstName: new FormControl(),
        lastName: new FormControl(),
        email: new FormControl(),
        sendCatalog: new FormControl(true)
  });
}
```

Checklist: FormBuilder

Import FormBuilder

Inject the FormBuilder instance

Use that instance

```
ngOnInit(): void {
  this.customerForm = this.fb.group({
              firstName: '',
              lastName: '',
              email: '',
              sendCatalog: true
  });
}
```

Checklist: Angular Module

Import ReactiveFormsModule

Add ReactiveFormsModule to the imports array

```
@NgModule({
  imports: [
    BrowserModule,
    ReactiveFormsModule
  ],
  declarations: [
    AppComponent,
    CustomerComponent
  ],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Checklist: Template

Bind the form element to the FormGroup property

```
<form class="form-horizontal"
      (ngSubmit)="save()"
      [formGroup]="customerForm">
```

Bind each input element to its associated FormControl

```
<input class="form-control"
       id="firstNameId"
       type="text"
       placeholder="First Name (required)"
       formControlName="firstName" />
```

Summary

- The Component Class
- The Angular Module
- The Template
- Using `setValue` and `patchValue`
- Simplifying with FormBuilder