

Validation



Deborah Kurata

CONSULTANT | SPEAKER | AUTHOR | MVP | GDE

@deborahkurata | blogs.msmvps.com/deborahk/





Module Overview

Setting Built-in Validation Rules

Adjusting Validation Rules at Runtime

Custom Validators

Custom Validators with Parameters

Cross-field Validation

Creating the Root FormGroup

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



Creating the FormControls

```
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: [true]  
});
```



Setting Built-in Validation Rules

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

```
this.customerForm = this.fb.group({  
    firstName: ['',  
        [Validators.required, Validators.minLength(3)]],  
    sendCatalog: true  
});
```



Adjusting Validation Rules at Runtime

Sign Up!

First Name

First Name (required)

Last Name

Last Name (required)

Email

Email (required)

Phone

Phone

Send Notifications

☐ Email ☐ Text

☐ Send me your catalog

Save



Adjusting Validation Rules at Runtime

```
myControl.setValidators(Validators.required);
```

```
myControl.setValidators([Validators.required,  
                        Validators.maxLength(30)]);
```

```
myControl.clearValidators();
```

```
myControl.updateValueAndValidity();
```



Custom Validator

```
function myCustomValidator(c: AbstractControl):  
    {[key: string]: boolean} | null {  
    if (somethingIsWrong) {  
        return { 'myvalidator': true };  
    }  
    return null;  
}
```



Custom Validator

Sign Up!

First Name

First Name (required)

Last Name

Last Name (required)

Email

Email (required)

Phone

Phone

**Send
Notifications**

☒ Email ☐ Text

Rating

☐ Send me your catalog

Save



Custom Validator

```
function myCustomValidator(c: AbstractControl):  
    {[key: string]: boolean} | null {  
    if (somethingIsWrong) {  
        return { 'myvalidator': true };  
    }  
    return null;  
}
```



Custom Validator with Parameters

```
function myCustomValidator(param: any): ValidatorFn {  
    return (c: AbstractControl):  
        {[key: string]: boolean} | null => {  
        if (somethingIsWrong) {  
            return { 'myvalidator': true };  
        }  
        return null;  
    }  
}
```



Cross-field Validation

Sign Up!

First Name

First Name (required)

Last Name

Last Name (required)

Email

Email (required)

Confirm Email

Confirm Email (required)

Phone

Phone

Send Notifications

☒ Email ☐ Text

Rating

☐ Send me your catalog

Save



Cross-field Validation: Nested FormGroup

```
this.customerForm = this.fb.group({  
  firstName: ['', [Validators.required, Validators.minLength(3)]],  
  lastName: ['', [Validators.required, Validators.maxLength(50)]],  
  availability: this.fb.group({  
    start: ['', Validators.required],  
    end: ['', Validators.required]  
  })  
});
```

```
<div formGroupName="availability">  
  ...  
  <input formControlName="start" />  
  ...  
  <input formControlName="end" />  
</div>
```



Cross-field Validation

Sign Up!

First Name

First Name (required)

Last Name

Last Name (required)

Email

Email (required)

Confirm Email

Confirm Email (required)

Phone

Phone

Send Notifications

☒ Email ☐ Text

Rating

☐ Send me your catalog

Save



Cross-field Validation: Custom Validator

```
function dateCompare(c: AbstractControl):  
    {[key: string]: boolean} | null {  
    let startControl = c.get('start');  
    let endControl = c.get('end');  
    if (startControl.value !== endControl.value) {  
        return { 'match': true };  
    }  
    return null;  
}
```

```
this.customerForm = this.fb.group({  
    firstName: ['', [Validators.required, Validators.minLength(3)]],  
    lastName: ['', [Validators.required, Validators.maxLength(50)]],  
    availability: this.fb.group({  
        start: ['', Validators.required],  
        end: ['', Validators.required]  
    }, { validator: dateCompare })  
});
```



Checklist: Setting Built-in Validation Rules



Import Validators

Pass in the validator or array of validators

```
ngOnInit(): void {  
  this.customerForm = this.fb.group({  
    firstName: ['', Validators.required],  
    lastName: ['', [Validators.required,  
                    Validators.maxLength(30)]],  
    email: '',  
    sendCatalog: true  
  });  
}
```



Checklist: Adjusting Validation Rules



Determine when to make the change

Use `setValidators` or `clearValidators`

Call `updateValueAndValidity`

```
setNotification(notifyVia: string): void {  
    const p = this.myForm.get('phone');  
    if (notifyVia === 'text') {  
        p.setValidators(Validators.required);  
    } else {  
        p.clearValidators();  
    }  
    p.updateValueAndValidity();  
}
```



Checklist: Custom Validators



Build a validator function

```
function myValidator(c: AbstractControl):  
    {[key: string]: boolean} | null {  
    if (somethingIsWrong) {  
        return { 'thisValidator': true };  
    }  
    return null;  
}
```

Use it like any other validator

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

Checklist: Custom Validators with Parameters



Wrap the validator function in a factory function

```
function myValidator(param: any): ValidatorFn {  
  return (c: AbstractControl):  
    {[key: string]: boolean} | null => {  
    if (c.value === param) {  
      return { 'thisvalidator': true };  
    }  
    return null;  
  }  
}
```

Use it like any other validator

```
this.customerForm = this.fb.group({  
  firstName: ['', myValidator('test')]  
});
```

Checklist: Cross-field Validation: FormGroup



Create a nested FormGroup

Add FormControls to that FormGroup

```
this.customerForm = this.fb.group({  
  name: ['', Validators.required],  
  availability: this.fb.group({  
    start: ['', Validators.required],  
    end: ['', Validators.required]  
  })  
});
```

Update the HTML

```
<div formGroupName="availability">  
  <input formControlName="start" />  
  <input formControlName="end" />  
</div>
```



Checklist: Cross-field Validation: Validator







Build a custom validator

```
function dateCompare(c: AbstractControl) {  
  if (c.get('start').value !==  
      c.get('end').value) {  
    return { 'match': true };  
  }  
  return null;  
}
```

Use the validator

```
this.customerForm = this.fb.group({  
  name: ['', Validators.required],  
  availability: this.fb.group({  
    start: [null, Validators.required],  
    end: [null, Validators.required]  
  }), { validator: dateCompare })  
});
```



Summary

Setting Built-in Validation Rules

Adjusting Validation Rules at Runtime

Custom Validators

Custom Validators with Parameters

Cross-field Validation