

ViewChild and ViewChildren



Deborah Kurata

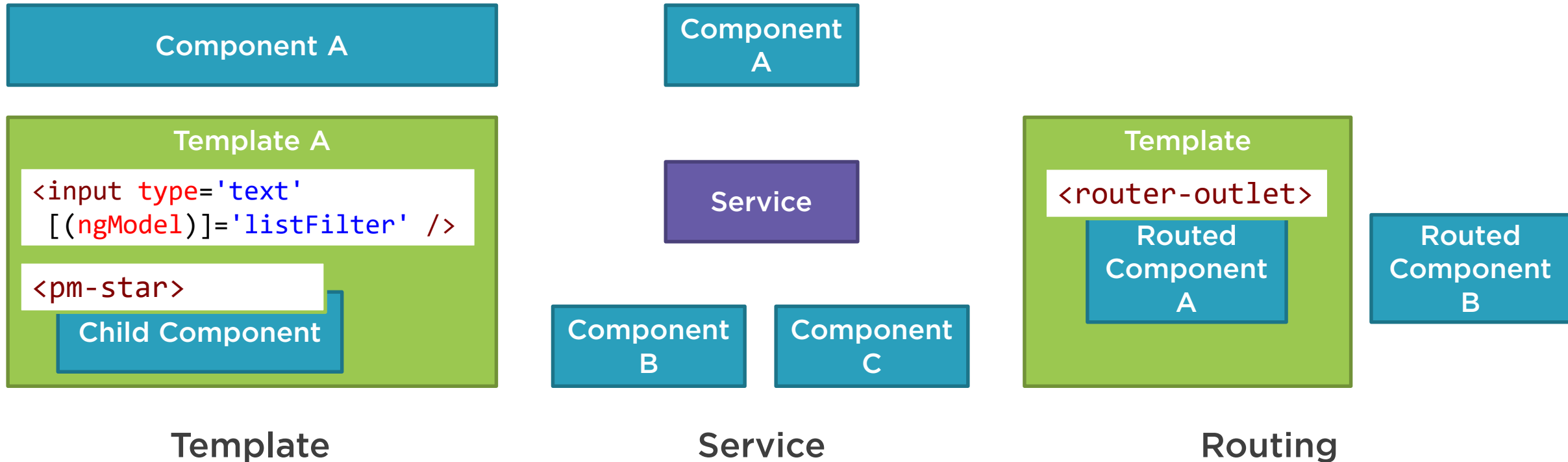
CONSULTANT | SPEAKER | AUTHOR | MVP | GDE

@deborahkurata | blogs.msmvps.com/deborahk/





Component Communication



Module Overview



ViewChild

ViewChildren

ViewChild and Angular Forms

- valueChanges Observable

ViewChild and ngIf



Could you set
your focus?

Sure!
Setting my
focus.

Component
A

Template
A



Is your form
currently valid?

OK, I'll save.

Yep!

Component
A

Template
A



Getting a Reference

DOM

```
let divElement = document.getElementById('divElementId');
```

Decorator

```
@ViewChild('divElementVar') divElementRef;
```



ViewChild

Angular Directive

```
@ViewChild(NgModel) filterInput: NgModel;
```

```
<input type='text' [(ngModel)]='listFilter' />
```

Custom Directive / Child Component

```
@ViewChild(StarComponent) star: StarComponent;
```

```
<pm-star [rating]='product.starRating'></pm-star>
```

Template Reference Variable

```
@ViewChild('divElementVar') divElementRef: ElementRef;
```

```
<div #divElementVar>{{pageTitle}}</div>
```



Demo



ViewChild and Accessing the Native Html Element



Considerations When Using nativeElement

**Using nativeElement ->
directly accessing the DOM**

Tightly coupled to the browser

May not be able to use server-side rendering or web workers

```
if (this.filterElementRef.nativeElement) {  
    this.filterElementRef.nativeElement.focus();  
}
```



ViewChildren

```
@ViewChildren('divElementVar')  
divElementRefs: QueryList<ElementRef>;
```

Differences:

- Returns a QueryList of element or directive references
- Tracks changes in the DOM

```
this.divElementRefs.changes.subscribe(() => {  
    // Code here  
})
```

ViewChildren

Angular Directive

```
@ViewChildren(NgModel) inputs: QueryList<NgModel>;
```

Custom Directive / Child Component

```
@ViewChildren(StarComponent) stars: QueryList<StarComponent>;
```

Template Reference Variable

```
@ViewChildren('divElementVar')  
divElementRefs: QueryList<ElementRef>;
```

Template Reference Variables

```
@ViewChildren('filterElement, nameElement')  
divElementRefs: QueryList<ElementRef>;
```



Demo



ViewChildren



Notifying the Component of User Changes



Two-way binding, the long way

Getter and setter

valueChanges observable

ViewChild and Angular Forms

Template

```
<input type='text'  
      [(ngModel)]='listFilter' />
```

Component

```
@ViewChild(NgModel) filterInput: NgModel;
```

```
this.filterInput.valueChanges.subscribe(  
  () => this.performFilter(this.listFilter)  
);
```



Angular Forms

Template-driven

- Angular creates the form data structures
- Based on info in the template
- Access reference with ViewChild

Reactive

- We create the form data structures
- Defined in the component class
- No need for ViewChild

```
this.filterInput.valueChanges.subscribe(  
  () => this.performFilter(this.listFilter)  
);
```



For More Information

Angular Forms

by Mark Zamoyta

Angular Reactive Forms

by Deborah Kurata



Template-driven Forms / No Form

```
<form (ngSubmit)="saveProduct()">
```

```
▼ NgForm {submitted: false, _d
  control: (...)
  ▼ controls: Object
    ► description: FormControl
    ► productCode: FormControl
    ► productName: FormControl
    ► __proto__: Object
  dirty: false
  disabled: (...)
  enabled: (...)
  errors: null
  ► form: FormGroup {validator:
  ► formDirective: NgForm
    invalid: (...)
  ► ngSubmit: EventEmitter {_is
    path: (...)
    pending: (...)
    pristine: (...)
    status: (...)
    statusChanges: (...)
    submitted: false
    touched: (...)
    untouched: (...)
    valid: (...)
    value: (...)
    valueChanges: (...)
```

```
<input type='text'
      [(ngModel)]= 'listFilter' />
```

```
▼ NgModel {_parent: null,
  asyncValidator: (...)
  ► control: FormControl
  dirty: (...)
  disabled: (...)
  enabled: (...)
  errors: (...)
  formDirective: null
  invalid: (...)
  model: undefined
  name: null
  path: (...)
  pending: (...)
  pristine: (...)
  status: (...)
  statusChanges: (...)
  touched: (...)
  untouched: (...)
  ► update: EventEmitter
    valid: (...)
    validator: (...)
    value: (...)
  ► valueAccessor: Default
    valueChanges: (...)
    viewModel: undefined
```



Demo



ViewChild and Angular Forms



ViewChild and *ngIf

```
<div *ngIf='products'>
  <div>Filter by:</div>
  <div>
    <input type='text' [(ngModel)]='listFilter' />
  </div>
</div>
```

```
@ViewChild(NgModel) filterInput: NgModel;
```

```
ngAfterViewInit(): void {
  this.filterInput.valueChanges.subscribe(
    () => this.performFilter(this.listFilter)
  );
}
```

```
@ViewChild(NgModel)
set filterInput(value: NgModel) {
  this._filterInput = value;
}
```



Demo



ViewChild and ngIf



Guidelines & Summary: ViewChild and ViewChildren



ViewChild/ViewChildren: Html Element

```
@ViewChild('divElementVar') divElementRef: ElementRef;
```

Plus:

- Provides a `nativeElement` property
- Access any Html element properties
- Call any Html element methods

Caveats:

- ViewChild reference not reliably available until `AfterViewInit`
- ViewChild reference not available if the element is not in the DOM
- Does not work with server-side rendering or web workers

```
if (this.filterElementRef.nativeElement) {  
    this.filterElementRef.nativeElement.focus();  
}
```



ViewChild/ViewChildren: Angular Directive

```
@ViewChild(NgModel) filterInput: NgModel;
```

Plus:

- Provides reference to the directive's data structures
- Access any properties

Caveats:

- ViewChild reference not reliably available until AfterViewInit
- ViewChild reference not available if the element is not in the DOM
- NgForm and NgModel data structures are read-only



Notifying the Component of User Changes



Two-way binding, the long way

Getter and setter

valueChanges observable

Subscribe to the valueChanges Observable

```
@ViewChild(NgModel) filterInput: NgModel;
```

```
this.filterInput.valueChanges.subscribe(  
  () => this.performFilter(this.listFilter)  
);
```

Plus:

- Favor this technique if using other NgModel information

Caveats:

- Watch out for ngIf
- Reference not reliably available until AfterViewInit



Summary



ViewChild

ViewChildren

ViewChild and Angular Forms

- valueChanges Observable

ViewChild and ngIf



Component Communication

