



ANGULAR  
**ARCHITECTS**  
INSIDE KNOWLEDGE

# Data Binding and OnPush

[ANGULARarchitects.io](https://ANGULARarchitects.io)

# Contents

- How does data binding work (underneath the covers)?
- Performance-Tuning with OnPush

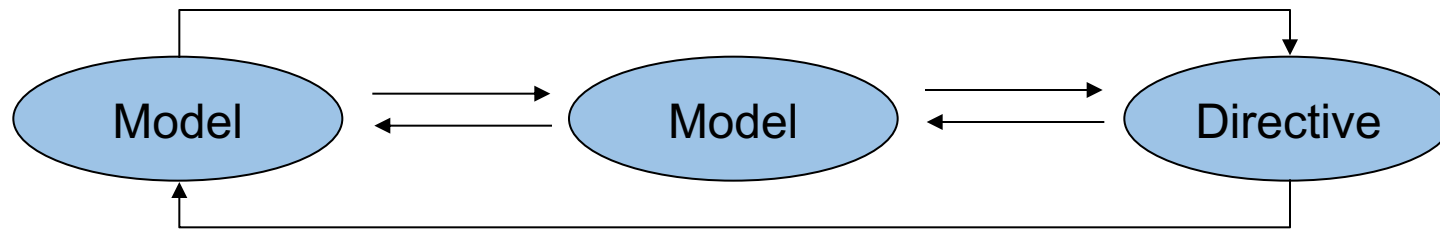


# Data Binding

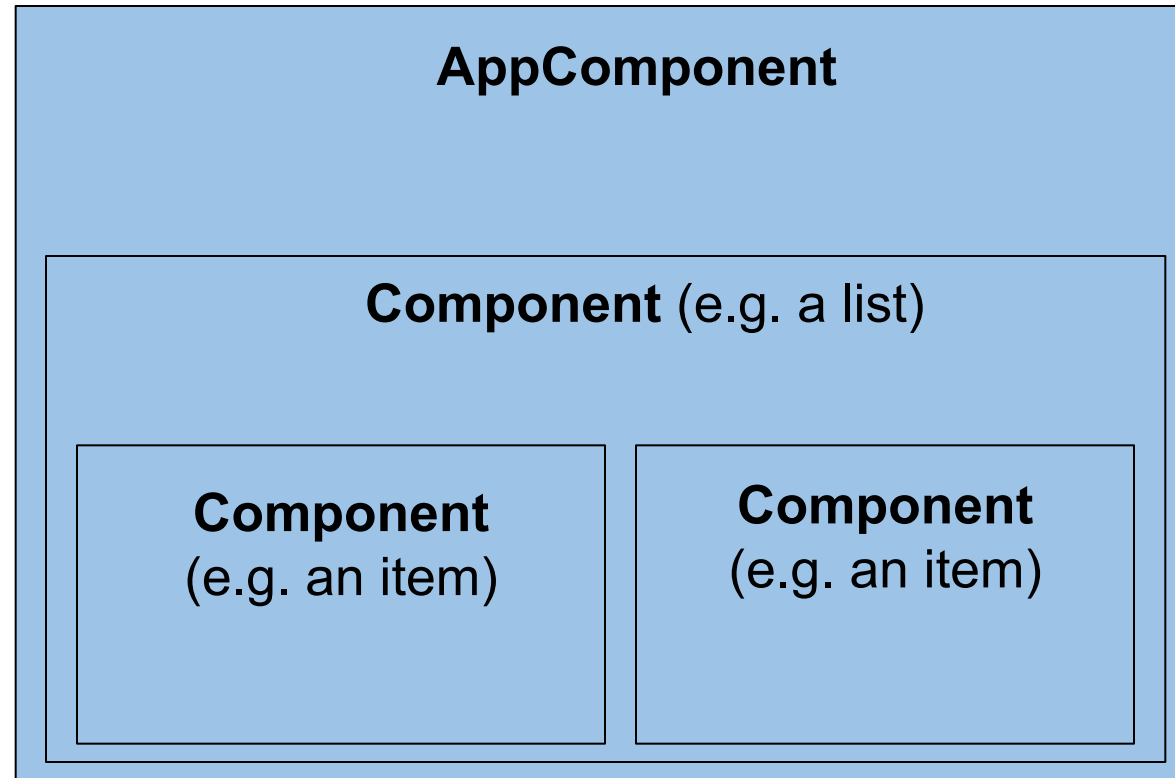


@ManfredSteyer

# Data Binding in AngularJS 1.x



# Component Tree in Angular 2+

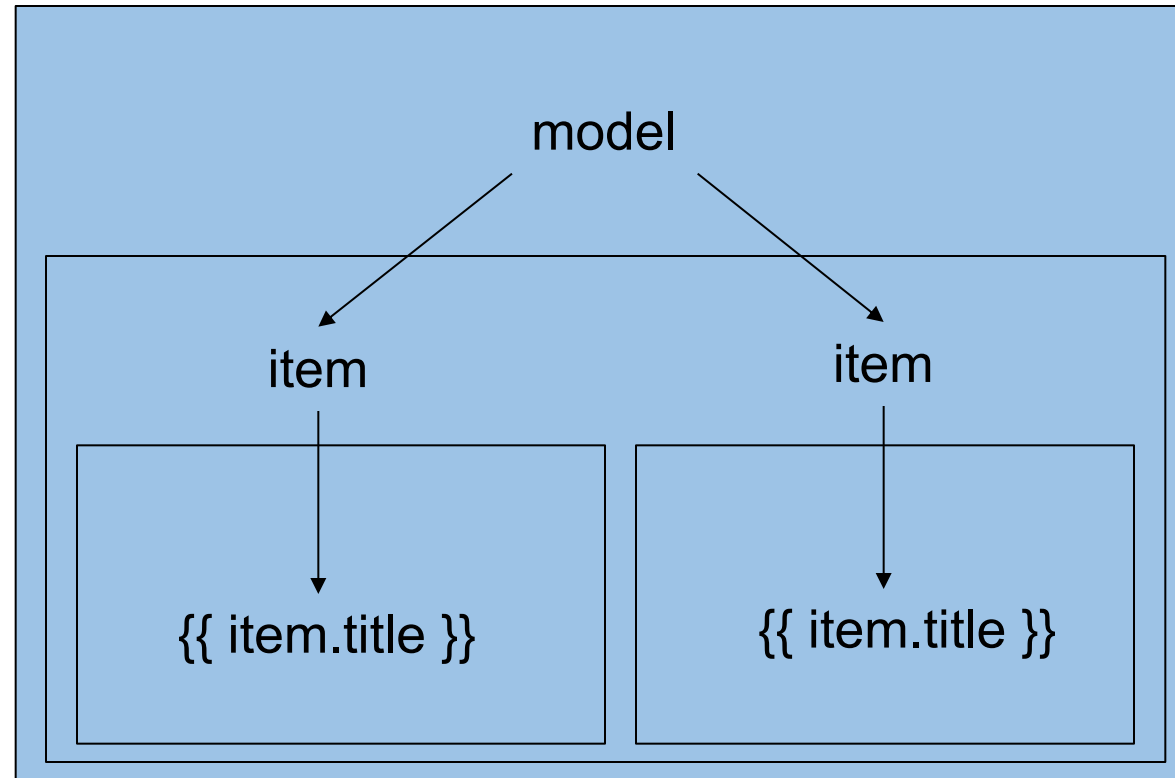


# Rules for Property-Bindings

- Data flows top/down
  - Parent can send data to children
  - Children **cannot** send data to parent
- Dependency graph is a tree
- Angular only needs one "digest"



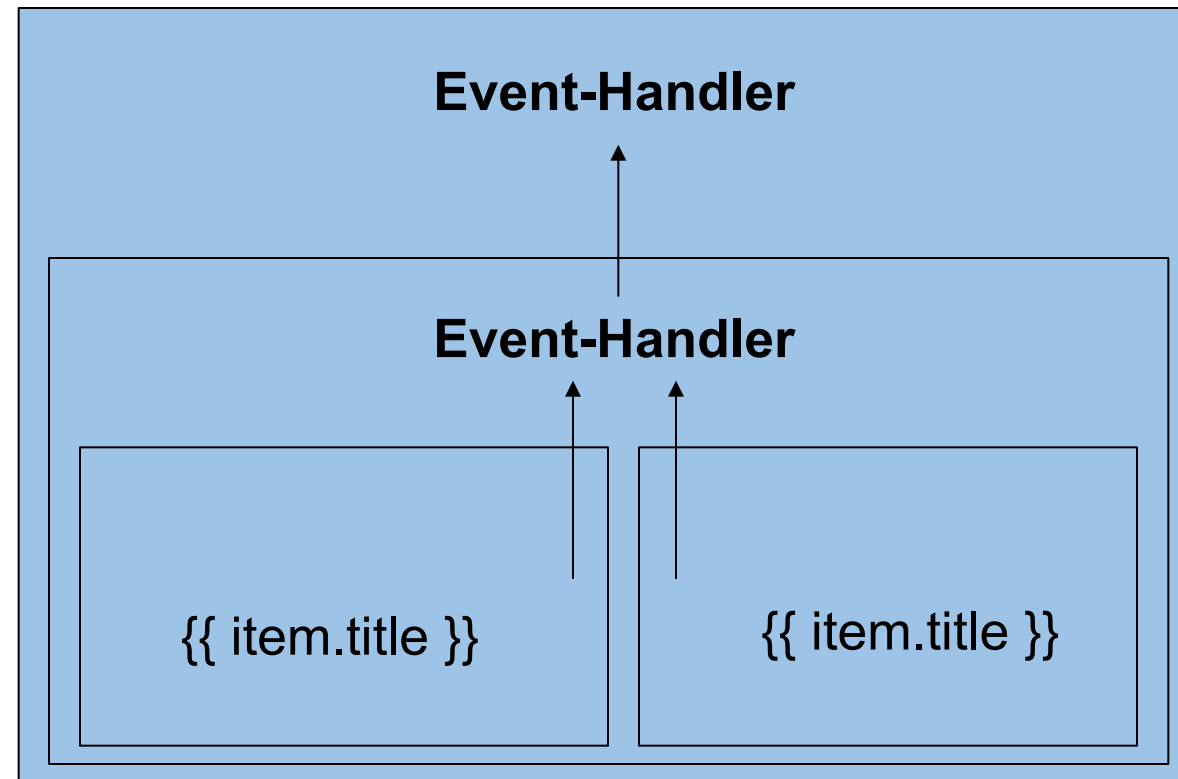
# Property Binding



[<http://victorsavkin.com/post/110170125256/change-detection-in-angular-2>]



# Event Bindings (One-Way, Bottom/Up)



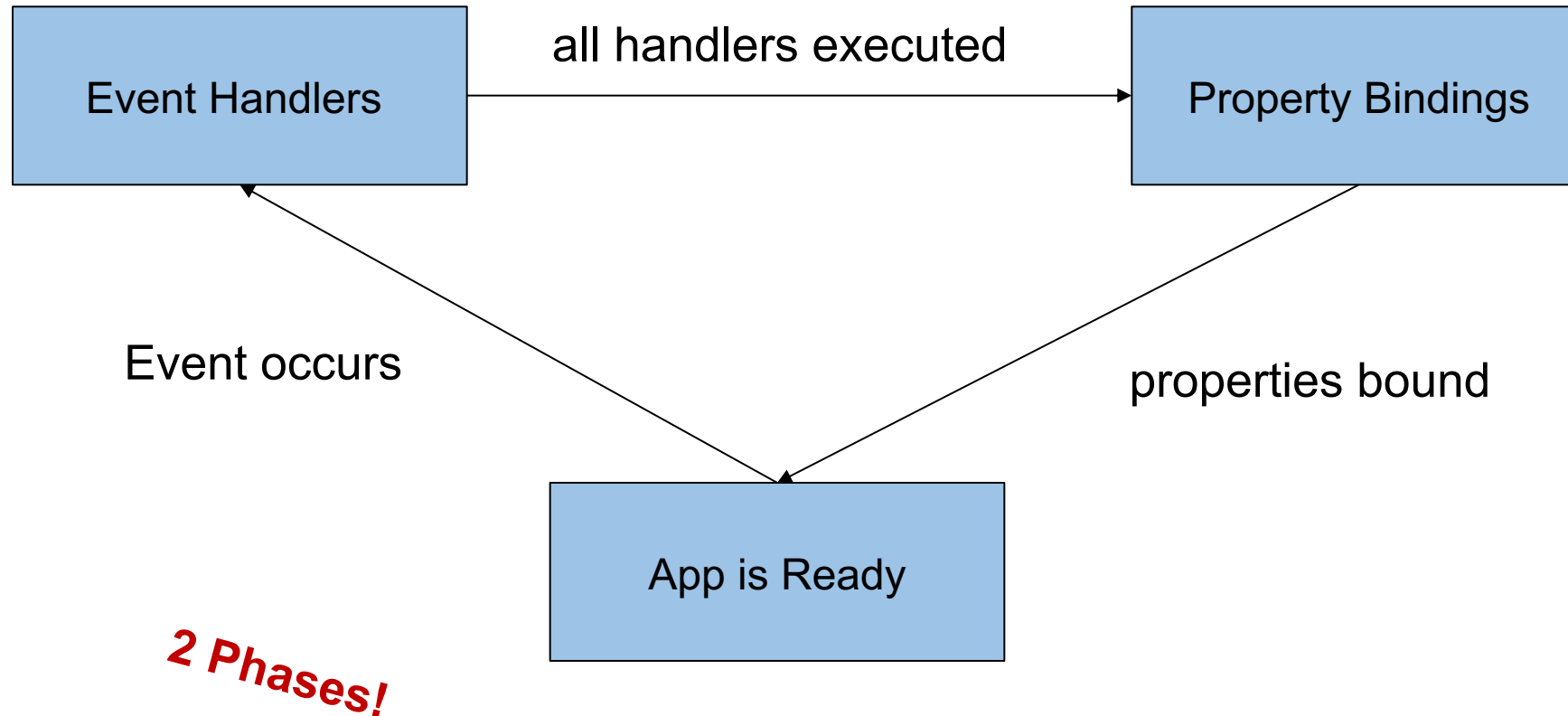


# Event Bindings (One-Way, Bottom/Up)

- Cheap: No "digest" needed!
- However: Events can change data → Property Binding



# Property- and Event-Bindings



# View

```
<button [disabled]="!von || !nach" (click)="search()">  
  Search  
</button>
```

```
<table>  
  <tr *ngFor="let flight of flights">  
    <td>{{flight.id}}</td>  
    <td>{{flight.date}}</td> ← - - - - - > <td [text-content]="flight.date"></td>  
    <td>{{flight.from}}</td>  
    <td>{{flight.to}}</td>  
    <td><a href="#" (click)="selectFlight(flight)">Select</a></td>  
  </tr>  
</table>
```



# DEMO



@ManfredSteyer

# Recap

- Property-Binding: One-Way; Top/Down
- Event-Binding: One-Way; Bottom/Up
- Two-Way-Binding?
- Two-Way = Property-Binding + Event-Binding

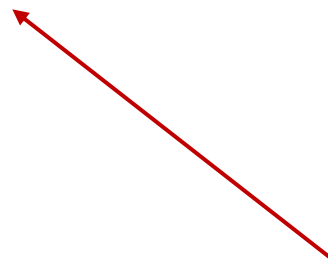


# Property and Event Bindings

```
<input [ngModel]="from" (ngModelChange)="update($event)">
```

# Property and Event Bindings

`<input [ngModel]="from" (ngModelChange)="from = $event">`



Property + *Change*

`<input [(ngModel)]="from">`



New Value



# DEMO



@ManfredSteyer



# Performance Tuning with OnPush



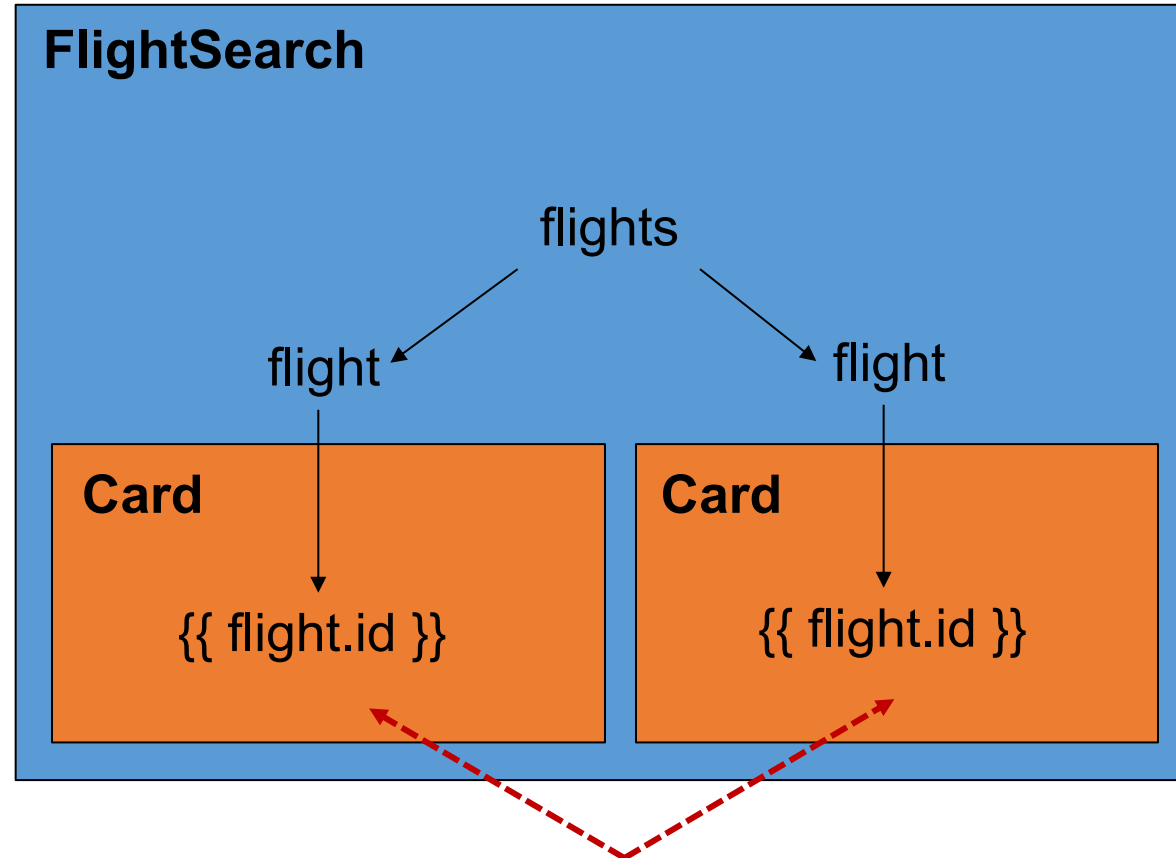
@ManfredSteyer

# DEMO



@ManfredSteyer

# OnPush



Angular just checks when “notified”



@ManfredSteyer

# "Notify" about change?

- Change bound data (@Input)
  - OnPush: Angular just compares the object reference!
  - e. g. `oldFlight === newFlight`
- Raise event within the component
- Notify a bound observable
  - `{{ flights$ | async }}`
- Trigger it manually
  - Don't do this at home ;-)
  - At least: Try to avoid this



# Activate OnPush

```
@Component({  
    [...]  
    changeDetection: ChangeDetectionStrategy.OnPush  
})  
export class FlightCard {  
    [...]  
    @Input() flight;  
}
```



# DEMO



@ManfredSteyer

# Summary

- Event Bindings → Property Bindings
- No cycles allowed!
- OnPush
- Immutables & Observables



LAB