



Global Knowledge®

In-company training QNH

Module 4

Angular Data binding

Peter Kassenaar –
info@kassenaar.com

WORLDWIDE LOCATIONS

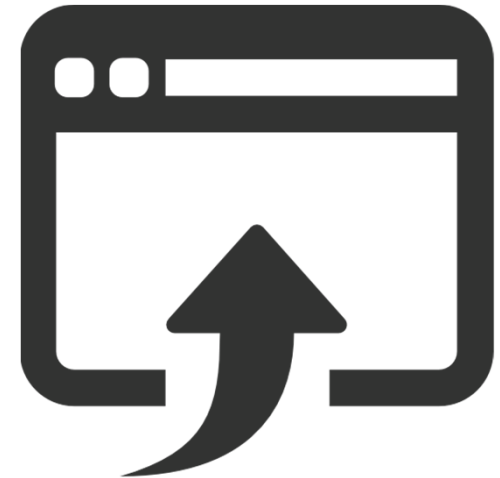
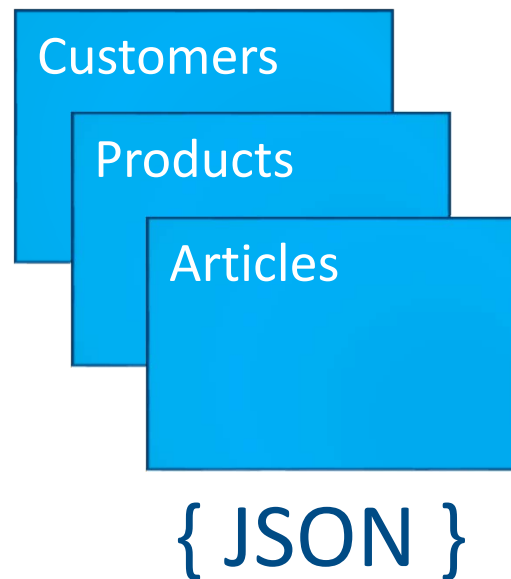
BELGIUM CANADA COLOMBIA DENMARK EGYPT FRANCE IRELAND JAPAN KOREA MALAYSIA MEXICO NETHERLANDS NORWAY QATAR
SAUDI ARABIA SINGAPORE SPAIN SWEDEN UNITED ARAB EMIRATES UNITED KINGDOM UNITED STATES OF AMERICA

Wat is databinding

Gegevens (data) tonen in de user interface

Data afkomstig uit:

- Controller / class
- Database
- User input
- Andere systemen



Declaratieve syntax

Nieuwe notatiewijzen in HTML-views/partials.

1. Simple data binding
2. Event binding
3. One-way data binding (Attribute binding)
4. Two-way data binding

Angular 1:

- Views zijn op zichzelf staande HTML-documenten. Krijgen via router in een app onderlinge samenhang

Angular 2:

- Views horen bij een bepaalde component.

1. Simple data binding syntax

Ongewijzigd ten opzichte van Angular 1. Dus nog steeds dubbele accolades:

```
<div>Stad: {{ city }}</div>
```

```
<div>Voornaam: {{ person.firstname }}</div>
```

Altijd: samenwerking met component/class

```
import {Component} from '@angular/core';  
  
@Component({  
  selector: 'hello-world',  
  template: `

# Hello Angular 2</h1> <h2>Mijn naam is : {{ name }}</h2> <h2>Mijn favoriete stad is : {{ city }}</h2> `, }) export class AppComponent { name = 'Peter Kassenaar'; city = 'Groningen' }


```

Of: properties via constructor

```
export class AppComponent {  
  name: string;  
  city: string;  
  
  constructor() {  
    this.name = 'Peter Kassenaar';  
    this.city = 'Groningen'  
  }  
}
```

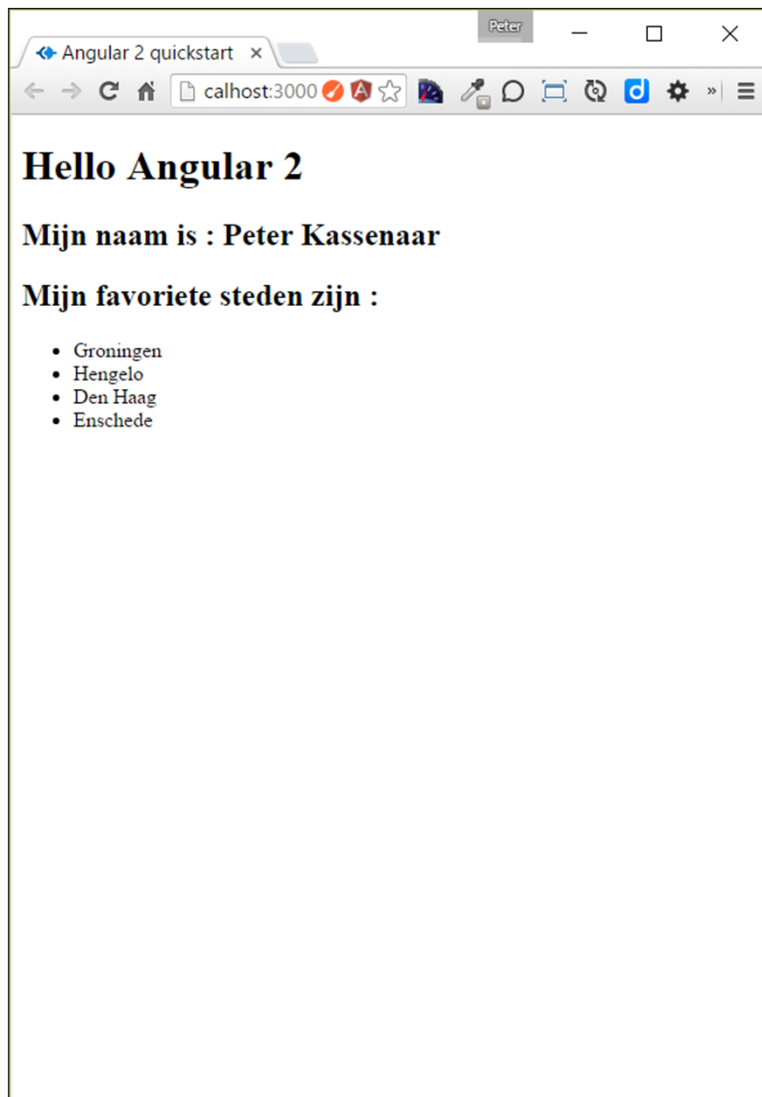
Vaak: persoonlijke voorkeur, of coding style/
organization preferences –
BEST PRACTICE: gebruik ngOnInit()

Binden via een lus: *ngFor

```
<h2>Mijn favoriete steden zijn :</h2>
<ul>
  <li *ngFor="let city of cities">{{ city }}</li> // vanaf .rc1
</ul>

// Class met properties, array met cities
export class AppComponent {
  name:string;
  cities:string[];

  constructor() {
    this.name = 'Peter Kassenaar';
    this.cities = ['Groningen', 'Hengelo', 'Den Haag', 'Enschede']
  }
}
```



Meer info:

<https://angular.io/docs/ts/latest/guide/displaying-data.html>

Model maken (als in: MVC)

Class met properties die wordt geëxporteerd:

```
export class City{  
  constructor(  
    public id: number,  
    public name: string,  
    public province: string,  
  ){ }  
}
```

Let op de shorthand notatie bij `public id : number :`

1. Maakt lokale parameter
2. Maakt publieke property met zelfde naam
3. Initialiseert property bij instantiëring van de class met `new`

Model gebruiken

1. Model-class importeren

```
import {City} from './city.model'
```

2. Component aanpassen

```
export class AppComponent {  
  name = 'Peter Kassenaar';  
  cities = [  
    new City(1, 'Groningen', 'Groningen'),  
    new City(2, 'Hengelo', 'Overijssel'),  
    new City(3, 'Den Haag', 'Zuid-Holland'),  
    new City(4, 'Enschede', 'Overijssel'),  
  ]  
}
```

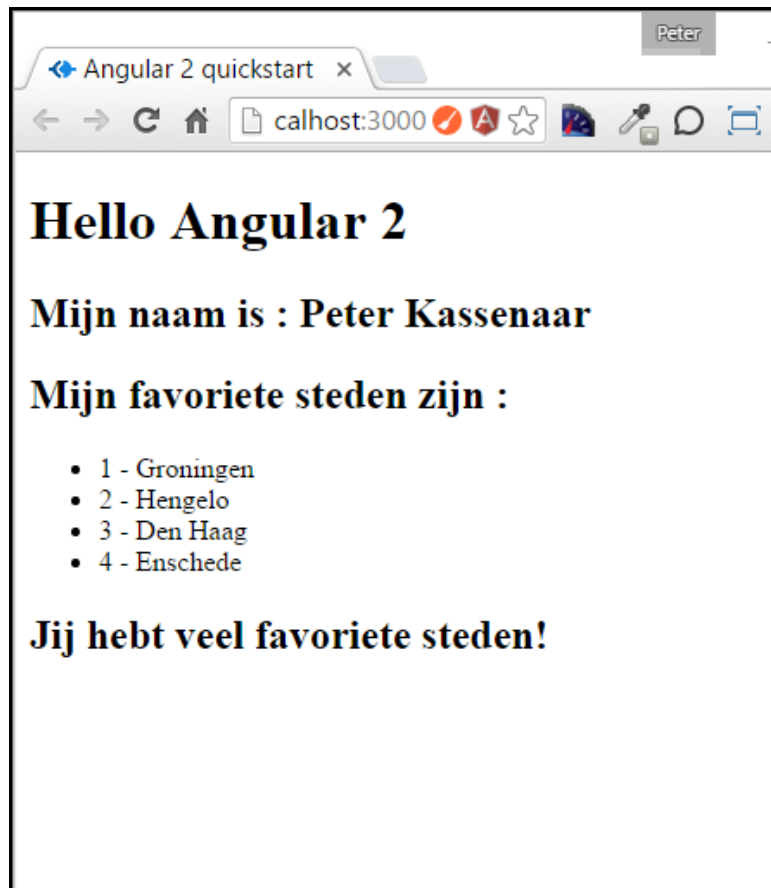
3. View aanpassen

```
<li *ngFor="let city of cities">{{ city.id}} - {{ city.name }}</li>
```

Voorwaardelijk tonen met *ngIf

Gebruik de directive *ngIf (let op het sterretje!)

```
<h2 *ngIf="cities.length > 3">Jij hebt veel favoriete steden!</h2>
```



Externe templates

Als je niet van inline HTML houdt:

```
@Component({  
  selector    : 'hello-world',  
  templateUrl: 'app/app.html'  
})
```



Bestand `app.html`

```
<!-- HTML in externe template -->  
<h1>Hello Angular 2</h1>  
<p>Dit is een externe template</p>  
<h2>Mijn naam is : {{ name }}</h2>  
<h2>Mijn favoriete steden zijn :</h2>  
...
```

Checkpoint

Simple data binding { { ... } }

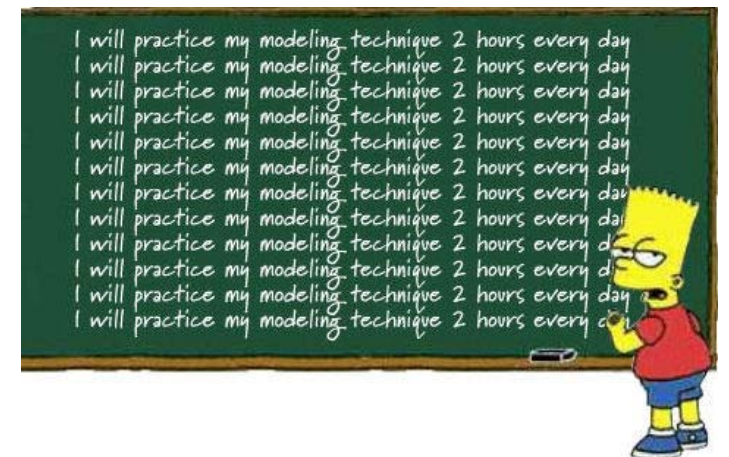
Properties van de class worden gebonden

Lussen en voorwaardelijke statement via `*ngFor` en `*ngIf`

Aanbevolen: werken met Model

Eventueel externe HTML-templates

Oefening....



User input en event binding

Reageren op mouse, keyboard,
hyperlinks en meer



Event binding syntax

Gebruik ronde haken voor events:

Angular 1:

```
<div ng-click="handleClick( )" >...</div>
```

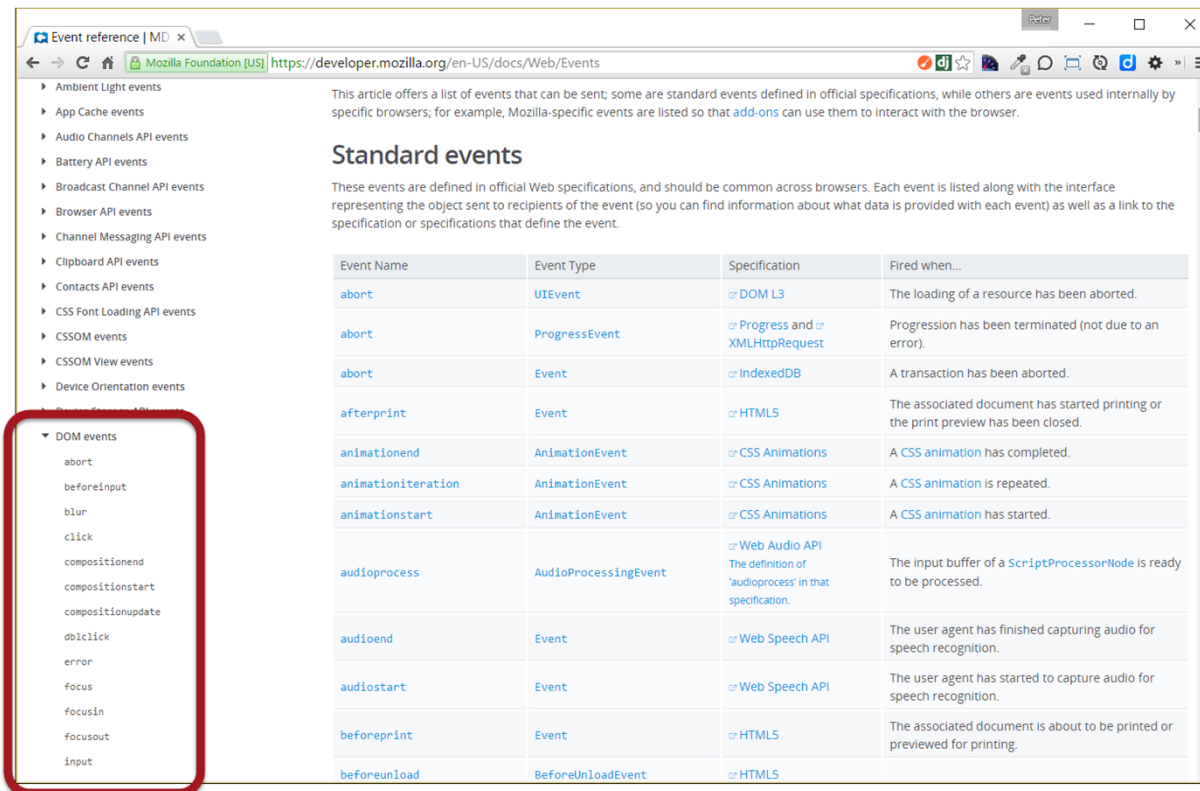
Angular 2:

```
<div (click)="handleClick( )" >...</div>
```

```
<div (blur)="onBlur( )" >...</div>
```

DOM-events

Angular2 kan naar *elk* DOM-event luisteren, zonder dat er een aparte directive voor nodig is:



This article offers a list of events that can be sent; some are standard events defined in official specifications, while others are events used internally by specific browsers; for example, Mozilla-specific events are listed so that [add-ons](#) can use them to interact with the browser.

Standard events

These events are defined in official Web specifications, and should be common across browsers. Each event is listed along with the interface representing the object sent to recipients of the event (so you can find information about what data is provided with each event) as well as a link to the specification or specifications that define the event.

Event Name	Event Type	Specification	Fired when...
abort	UIEvent	DOM L3	The loading of a resource has been aborted.
abort	ProgressEvent	Progress and XMLHttpRequest	Progression has been terminated (not due to an error).
abort	Event	IndexedDB	A transaction has been aborted.
afterprint	Event	HTML5	The associated document has started printing or the print preview has been closed.
animationend	AnimationEvent	CSS Animations	A CSS animation has completed.
animationiteration	AnimationEvent	CSS Animations	A CSS animation is repeated.
animationstart	AnimationEvent	CSS Animations	A CSS animation has started.
audioprocess	AudioProcessingEvent	Web Audio API The definition of 'audioprocess' in that specification.	The input buffer of a ScriptProcessorNode is ready to be processed.
audioend	Event	Web Speech API	The user agent has finished capturing audio for speech recognition.
audiostart	Event	Web Speech API	The user agent has started to capture audio for speech recognition.
beforeprint	Event	HTML5	The associated document is about to be printed or previewed for printing.
beforeunload	BeforeUnloadEvent	HTML5	

<https://developer.mozilla.org/en-US/docs/Web/Events>

Voorbeeld event binding

```
<!-- Event binding voor een button -->
```

```
<button class="btn btn-success"
```

```
  (click)="btnClick()">Ik ben een button</button>
```

```
export class AppComponent {
```

```
  ...
```

```
  counter: number = 0;
```

```
  btnClick(){
```

```
    alert('Je hebt ' + ++this.counter + ' keer geklikt');
```

```
  }
```

```
}
```



- Veel editors geven intellisense voor de beschikbare events
- In Angular taal:
 - Links van het isgelijkteken: *target of the binding*
 - Rechts van het isgelijkteken: *template expression*
 - Template expression wordt uitgevoerd in de *execution context* (= huidige class)

Event binding met \$event

```
<input type="text" class="input-lg" placeholder="Plaatsnaam..."  
      (keyup)="onKeyUp($event)"><br>  
<p>{{ txtKeyUp }}</p>
```

// 2. Binden aan keyUp-event in de textbox

```
onKeyUp(event:any){  
    this.txtKeyUp = event.target.value + ' - '  
  
}
```

Binding met local template variable

Declareer *local template variable* met # → Het hele element wordt doorgegeven aan de component

```
<input type="text" class="input-lg" placeholder="Plaatsnaam..."  
      #txtCity (keyup)="betterKeyUp()">  
<h3>{{ txtCity.value }}</h3>
```

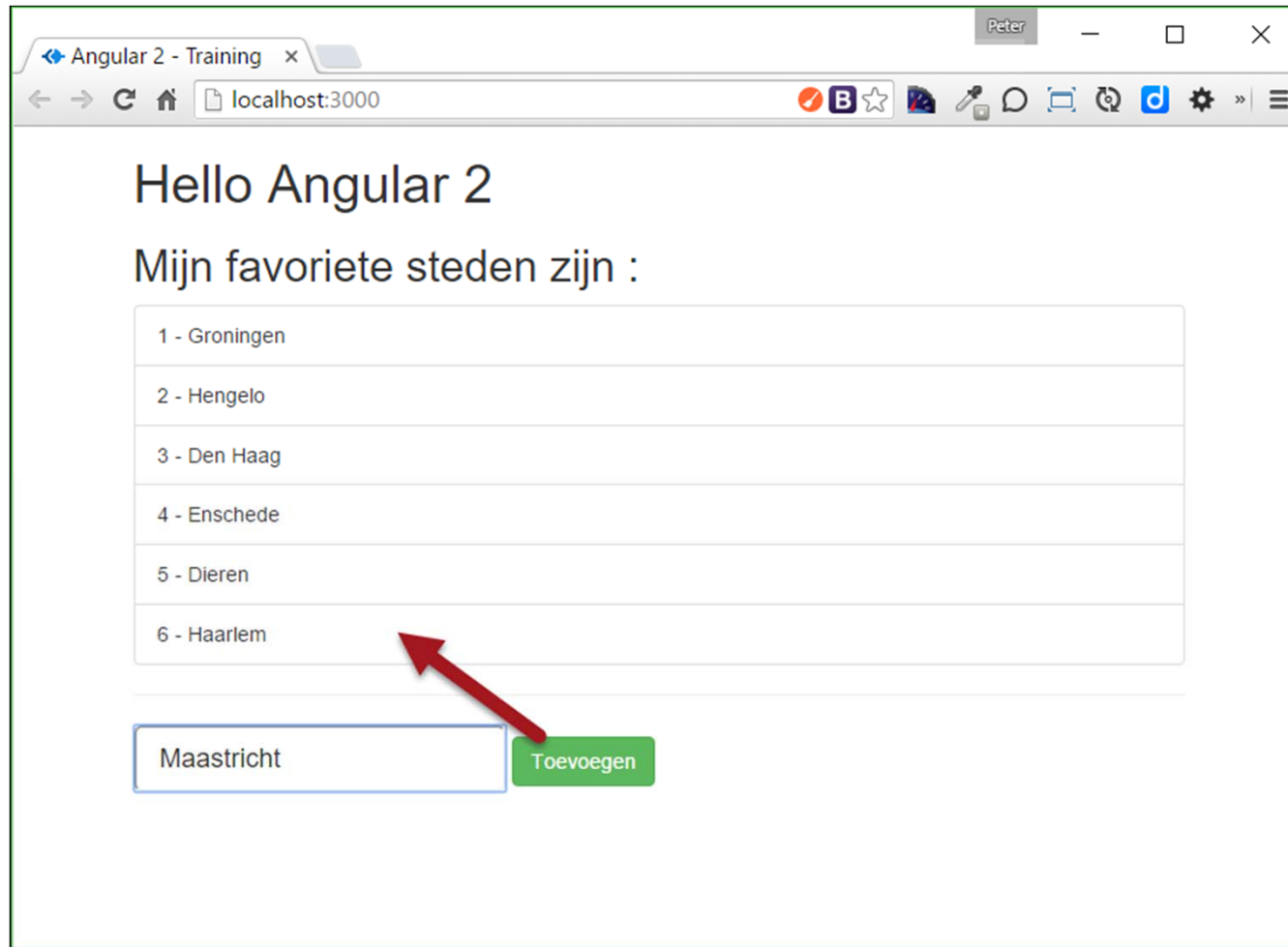
Class:

```
// 3. Binden aan keyUp-event via local template variable  
betterKeyUp(){  
    //... do nothing, for now  
}
```

Putting it all together...

```
<input type="text" class="input-lg" placeholder="Plaatsnaam..." #txtCity>
<button class="btn btn-success"
    (click)="addCity(txtCity)">Toevoegen
</button>
```

```
export class AppComponent {
    // Properties voor de component/class
    ...
    addCity(txtCity) {
        let newID    = this.cities.length + 1;
        let newCity = new City(newID, txtCity.value, 'Onbekend');
        this.cities.push(newCity);
        txtCity.value = '';
    }
}
```



Verder lezen/meer informatie: <https://angular.io/docs/ts/latest/guide/user-input.html>

Checkpoint

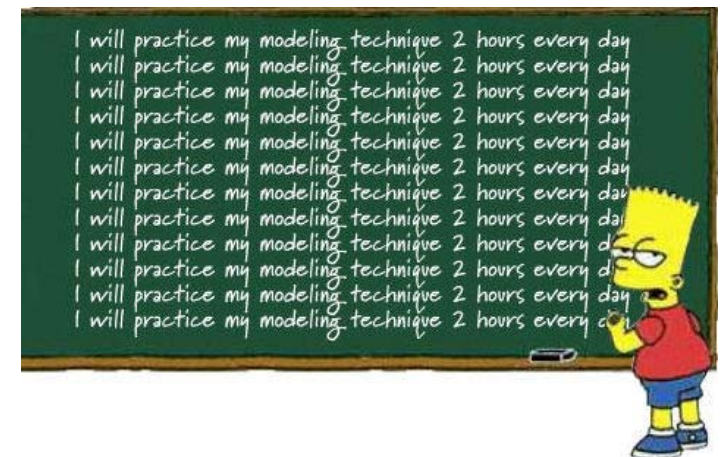
Event binding wordt aangegeven met `(eventName) = "..."`

Events worden afgehandeld door een event handler-functie in de component

Gebruik `#` om een local template variable te declareren.

Op deze manier zijn eenvoudige CRUD-operations te realiseren.

Oefening....



Attribute & property binding

Eigenschappen binden aan
HTML-attributen en DOM-properties



Attribute binding syntaxis

Rechtstreeks binden aan properties van HTML-elementen.

Ook wel: *one-way binding*.

Gebruik blokhaken syntaxis

Angular 1:

```
<div ng-hide="true|false">...</div>
```

Angular 2:

```
<div [hidden]="true">...</div>
```

Of :

```
<div [hidden]="person.hasEmail">...</div>
```

```
<div [style.background-color]=" 'yellow' ">...</div>
```

Voorbeeld attribute binding

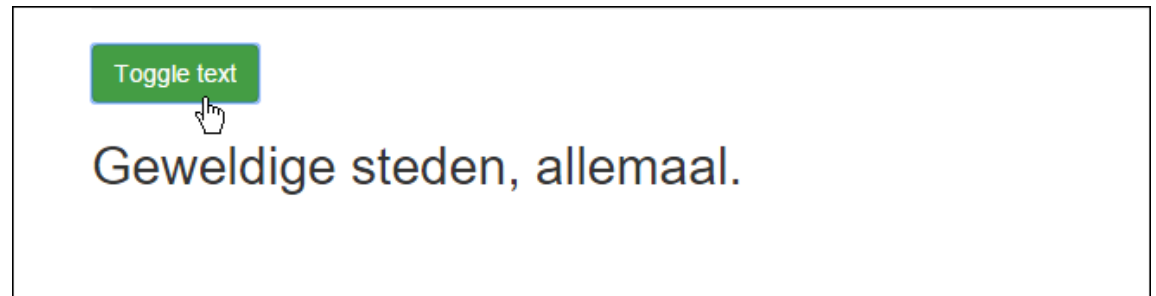
```
<!-- Attribute binding -->
```

```
<button class="btn btn-success" (click)="toggleText()">Toggle text</button>
```

```
<h2 [hidden]="textVisible">Geweldige steden, allemaal.</h2>
```

```
// attribuut toggelen: tekst zichtbaar/onzichtbaar maken.
```

```
toggleText(){  
  this.textVisible = !this.textVisible;  
}
```



Bijvoorbeeld...

```
<li *ngFor="let city of cities" class="list-group-item"
  (click)="updateCity(city)">
  {{ city.id }} - {{ city.name }}
</li>
```

```
export class AppComponent {
  // ...
  currentCity:City      = null;
  cityPhoto:string      = '';

  // Geselecteerde city updaten in de ui. Nieuw : ES6 String interpolation
  updateCity(city:City) {
    this.currentCity = city;
    this.cityPhoto    = `img/${this.currentCity.name}.jpg`;
  }
}
```

..\103-attributebinding\app\app-02.html en

..\app-02.component.ts

Hello Angular 2

Mijn favoriete steden zijn :

1 - Groningen

2 - Hengelo

3 - Den Haag

4 - Enschede



mijn stad: Groningen

Meer informatie: <https://angular.io/docs/ts/latest/guide/template-syntax.html#!#property-binding>

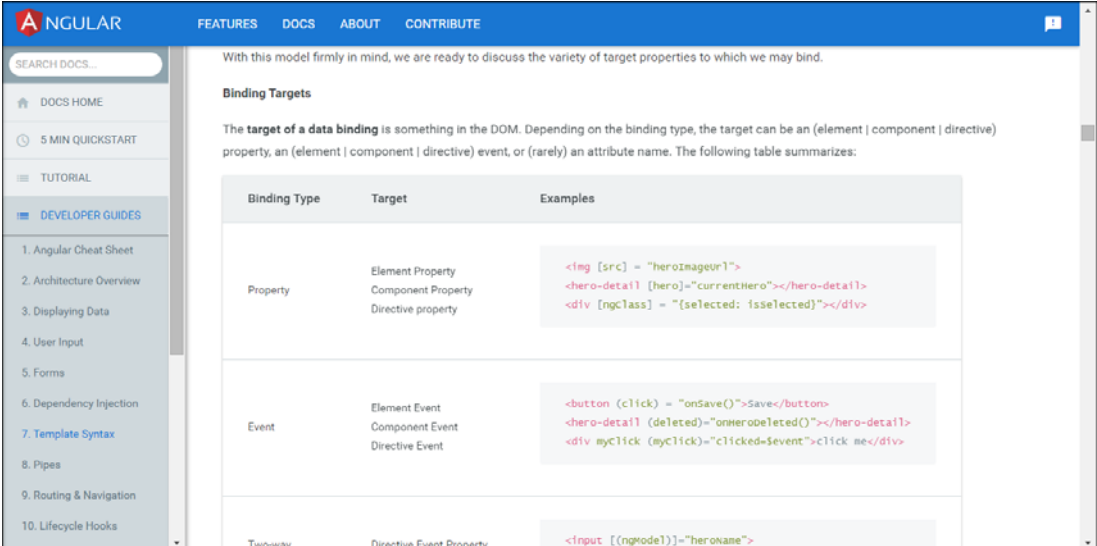
Meer binding-opties

Attribute binding en DOM-property binding

Class binding : [ngClass]

Style binding : [ngStyle]

<https://angular.io/docs/ts/latest/guide/template-syntax.html>



The screenshot shows the Angular documentation page for Template Syntax. The page has a blue header with the Angular logo and navigation links: FEATURES, DOCS, ABOUT, and CONTRIBUTE. A search bar is located in the top left. On the left side, there is a sidebar with a 'SEARCH DOCS...' input and a list of links: DOCS HOME, 5 MIN QUICKSTART, TUTORIAL, and DEVELOPER GUIDES. Under DEVELOPER GUIDES, there is a list of topics: 1. Angular Cheat Sheet, 2. Architecture Overview, 3. Displaying Data, 4. User Input, 5. Forms, 6. Dependency Injection, 7. Template Syntax (highlighted), 8. Pipes, 9. Routing & Navigation, and 10. Lifecycle Hooks. The main content area has a blue header with the text 'With this model firmly in mind, we are ready to discuss the variety of target properties to which we may bind.' Below this is the section 'Binding Targets'. It explains that the target of a data binding is something in the DOM and lists the following table summarizes: Binding Type, Target, and Examples. The table has three rows: Property, Event, and Two-way. The Property row lists Element Property, Component Property, and Directive property with examples like , <hero-detail [hero]="currentHero"></hero-detail>, and <div [ngClass] = "{selected: isSelected}"></div>. The Event row lists Element Event, Component Event, and Directive Event with examples like <button (click) = "onSave()">Save</button>, <hero-detail (deleted)="onherodeleted()"></hero-detail>, and <div myClick (myClick)="clicked=\$event">click me</div>. The Two-way row lists Directive Event Property with the example <input [(ngmodel)]="heroname">.

Binding Type	Target	Examples
Property	Element Property Component Property Directive property	<code></code> <code><hero-detail [hero]="currentHero"></hero-detail></code> <code><div [ngClass] = "{selected: isSelected}"></div></code>
Event	Element Event Component Event Directive Event	<code><button (click) = "onSave()">Save</button></code> <code><hero-detail (deleted)="onherodeleted()"></hero-detail></code> <code><div myClick (myClick)="clicked=\$event">click me</div></code>
Two-way	Directive Event Property	<code><input [(ngmodel)]="heroname"></code>

Two-way binding

User interface en logica
gelijktijdig updaten



Two way binding syntax

Is een tijdje weg geweest uit Angular 2, maar op veler verzoek toch teruggekeerd

Angular 1:

```
<input ng-model="person.firstName" />
```

Angular 2: de notatie is een beetje bizar:

```
<input [ (ngModel) ]="person.firstName" />
```

[(ngModel)] gebruiken

```
<input type="text" class="input-lg" [(ngModel)]="newCity" />  
<h2>{{ newCity }}</h2>
```

Dat is shorthand-notatie voor:

```
<!-- Two-way binding met uitgebreide syntaxis-->  
<input type="text" class="input-lg"  
      [value]="newCityExtended"  
      (input)="newCityExtended = $event.target.value" />  
<h2>{{ newCityExtended }}</h2>
```


FormsModule importeren

Vroeger maakte de Formulier-functionaliteit standaard deel uit van Angular.

Nu niet meer – apart importeren in `app.module.ts`!

```
import {FormsModule} from "@angular/forms";
```

...

```
imports      : [BrowserModule, FormsModule],
```

Data van View naar Controller

1. Via \$event
2. Via Local Template Variabele #NameVar
3. Via [(ngModel)] (met name in eenvoudige situaties, niet voor complexe formulieren)
4. HostBinding/HostListener (via @-decorators)

Binding cheat sheet

The screenshot shows the Angular.io website's 'ANGULAR CHEAT SHEET' page for Angular 2 for TypeScript. The page has a blue header with the Angular logo and navigation links: FEATURES, DOCS, ABOUT, and CONTRIBUTE. A search bar is located in the top left. On the left side, there is a sidebar with a 'DEVELOPER GUIDES' section containing a list of 13 topics, with '1. Angular Cheat Sheet' selected. The main content area has a blue header with 'ANGULAR CHEAT SHEET' and a dropdown menu set to 'Angular 2 for TypeScript'. Below this, a notice states: 'This cheat sheet is provisional and may change. Angular 2 is currently in Beta.' The main content is titled 'Angular for TypeScript Cheat Sheet (v2.0.0-beta.0)'. It features two sections: 'Bootstrapping' and 'Template syntax'. The 'Bootstrapping' section shows a code snippet: `import {bootstrap} from 'angular2/angular2';` followed by `bootstrap(MyAppComponent, [MyService, provide(...)]);` and an explanation: 'Bootstraps an application with MyAppComponent as the root component and configures the DI providers.' The 'Template syntax' section shows three examples of Angular template syntax: `<input [value]="firstName">` (binds property value to firstName), `<div [attr.role]="myAriaRole">` (binds attribute role to myAriaRole), and `<div [class.extra-sparkle]="isDelightful">` (binds the presence of the CSS class extra-sparkle to isDelightful).

ANGULAR

FEATURES DOCS ABOUT CONTRIBUTE

SEARCH DOCS...

DOCS HOME

5 MIN QUICKSTART

TUTORIAL

DEVELOPER GUIDES

- 1. Angular Cheat Sheet
- 2. Architecture Overview
- 3. Displaying Data
- 4. User Input
- 5. Forms
- 6. Dependency Injection
- 7. Template Syntax
- 8. Pipes
- 9. Routing & Navigation
- 10. Lifecycle Hooks
- 11. Attribute Directives
- 12. Structural Directives
- 13. Hierarchical Injectors

ANGULAR CHEAT SHEET

Angular 2 for TypeScript

This cheat sheet is provisional and may change. Angular 2 is currently in Beta.

Angular for TypeScript Cheat Sheet (v2.0.0-beta.0)

Bootstrapping	
<code>bootstrap(MyAppComponent, [MyService, provide(...)]);</code>	Bootstraps an application with MyAppComponent as the root component and configures the DI providers.

Template syntax	
<code><input [value]="firstName"></code>	Binds property <code>value</code> to the result of expression <code>firstName</code> .
<code><div [attr.role]="myAriaRole"></code>	Binds attribute <code>role</code> to the result of expression <code>myAriaRole</code> .
<code><div [class.extra-sparkle]="isDelightful"></code>	Binds the presence of the CSS class <code>extra-sparkle</code> on

<https://angular.io/docs/ts/latest/guide/cheatsheet.html>

Ingebouwde directives

Veel directives konden vervallen door de nieuwe syntaxis. Er zijn er nog maar weinig over.

Directives die het DOM manipuleren: herkenbaar aan sterretje/asterisk

```
<div *ngFor="let person of Persons">...</div>
```

```
<div *ngIf="showDiv">...</div>
```

```
<div [ngClass]="setClasses()">...</div>
```

```
<div [ngStyle]="setStyles()">...</div>
```

Samenvatting...

Databinding is in Angular 2 vernieuwd

Leer werken met de nieuwe notatie voor DOM- en Attribute binding, event binding en two-way binding

Pas altijd de Component en de bijbehorende View aan.

Veel concepten komen overeen, de uitwerking is totaal nieuw, in vergelijking met Angular 1