

Please evaluate
this talk via the
mobile app!



THE ROAD TO ANGULAR 2.0



ABOUT ME

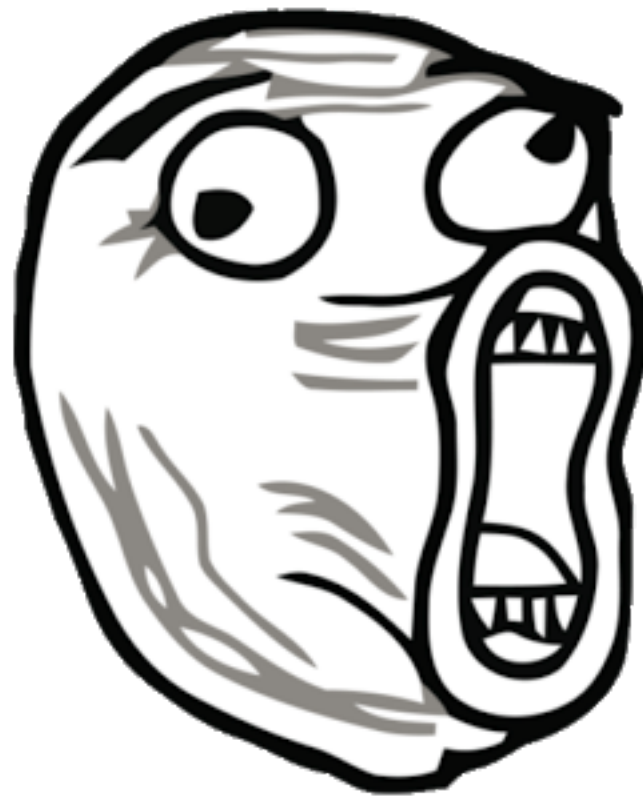
- Angular 1.x apps
- iOS apps
- REST API's in Java

Maarten Hus, Developer @

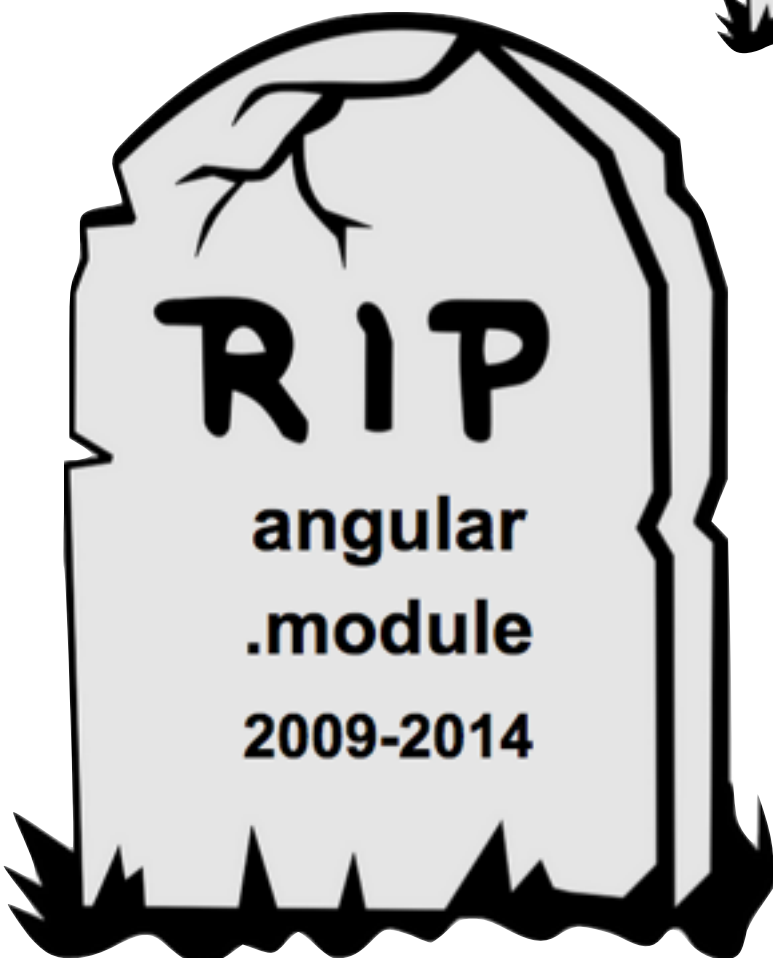


WHY THIS TALK?

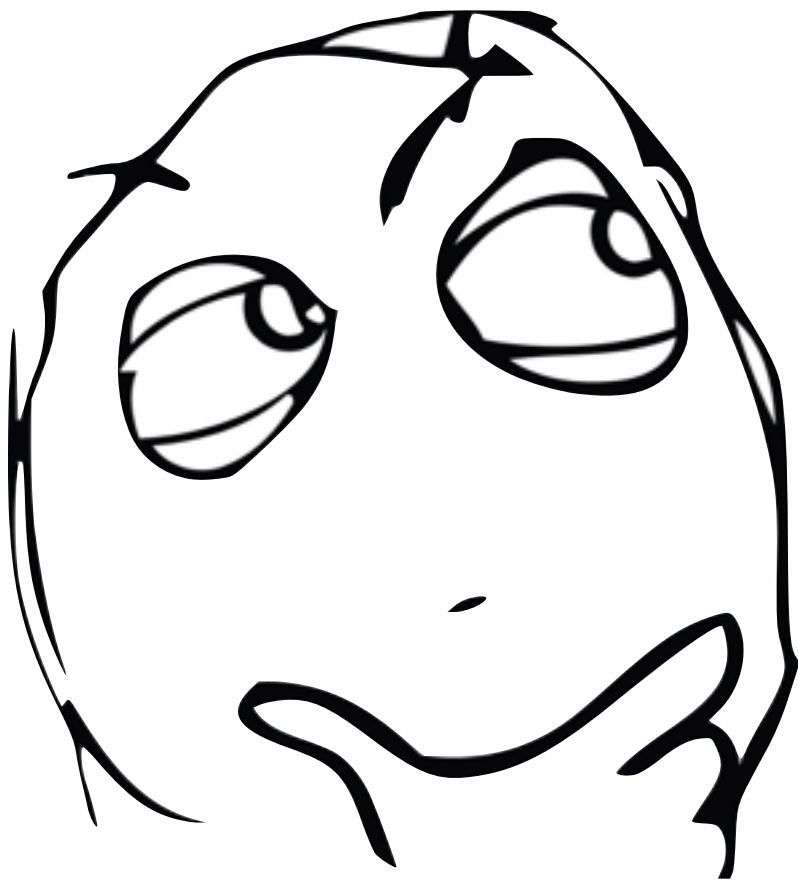
WAIT WHAT?!



GRAVEYARD



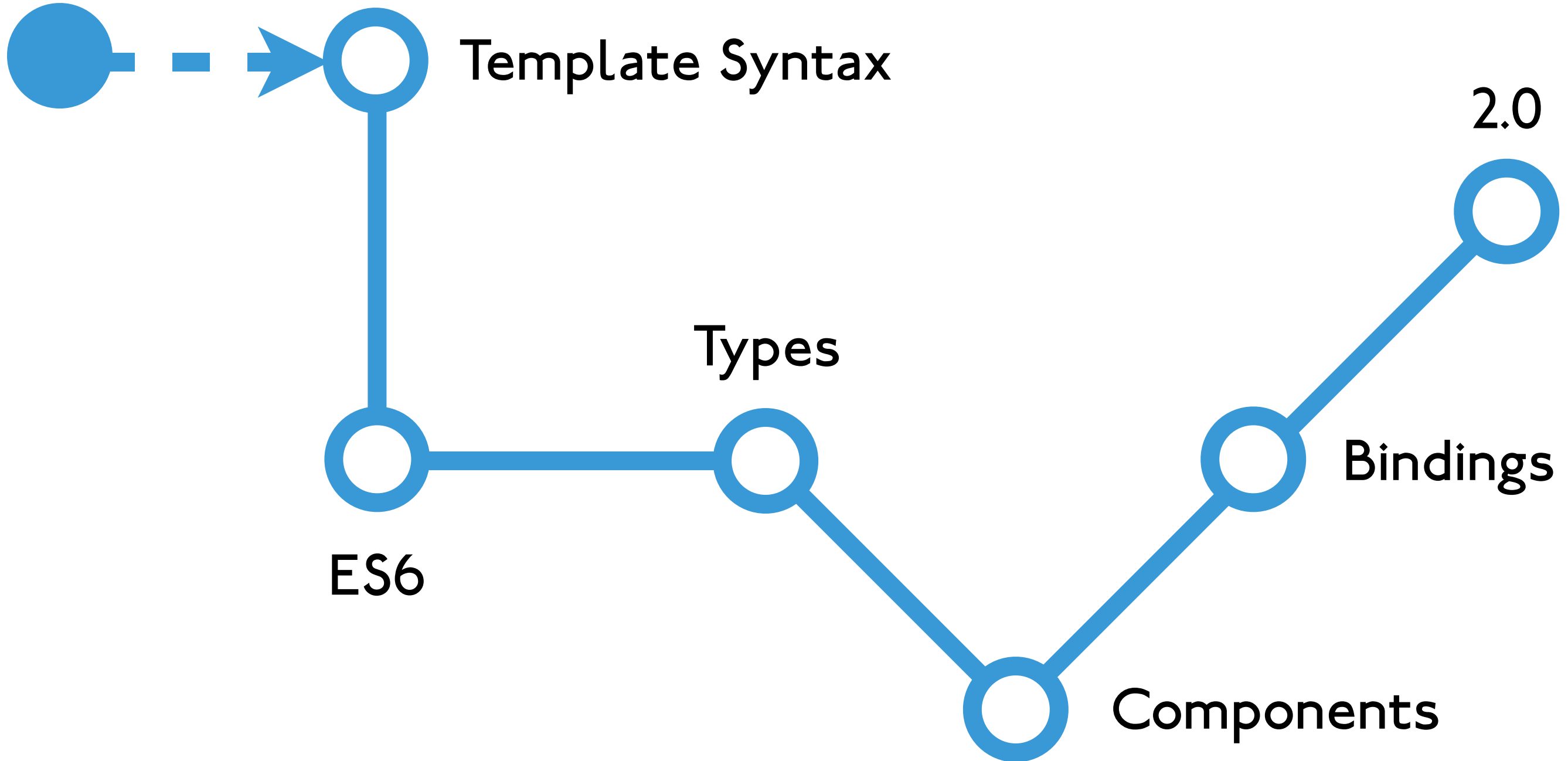
WHY?



DISCLAIMER

THE ROAD

1.x



TEMPLATE SYNTAX

Angular 2.0's template syntax is radically different from 1.x's.

DIFFERENCES

1.x

```
<span>Username: {{user.username}}</span>  
  
<button ng-click="upvote(user)">+1</button>
```

2.0

```
<span>Username: {{user.username}}</span>  
<img [src]="{{ user.imageUrl }}" />  
<button (click)="upvote(user)">+1</button>
```

DIFFERENCES

1.x

```
<span>Username: {{user.username}}</span>  
  
<button ng-click="upvote(user)">+1</button>
```

2.0

```
<span>Username: {{user.username}}</span>  
<img [src]="{{ user.imageUrl }}" />  
<button (click)="upvote(user)">+1</button>
```

DIFFERENCES

1.x

```
<span>Username: {{user.username}}</span>  
  
<button ng-click="upvote(user)">+1</button>
```

2.0

```
<span>Username: {{user.username}}</span>  
<img [src]="{{ user.imageUrl }}" />  
<button (click)="upvote(user)">+1</button>
```

DIFFERENCES

1.x

```
<span>Username: {{user.username}}</span>  
  
<button ng-click="upvote(user)">+1</button>
```

2.0

```
<span>Username: {{user.username}}</span>  
<img [src]="{{ user.imageUrl }}" />  
<button (click)="upvote(user)">+1</button>
```

2.0 SYNTAX

[] => property binding, is an expression

() => event, is a statement, caused by the user.

NO NEED FOR

*ng-bind, ng-src, ng-class, ng-href,
ng-show, ng-hide, ng-click, ng-enabled,
ng-disabled, ng-blur, ng-keyup, ng-
mouse-down etc etc*

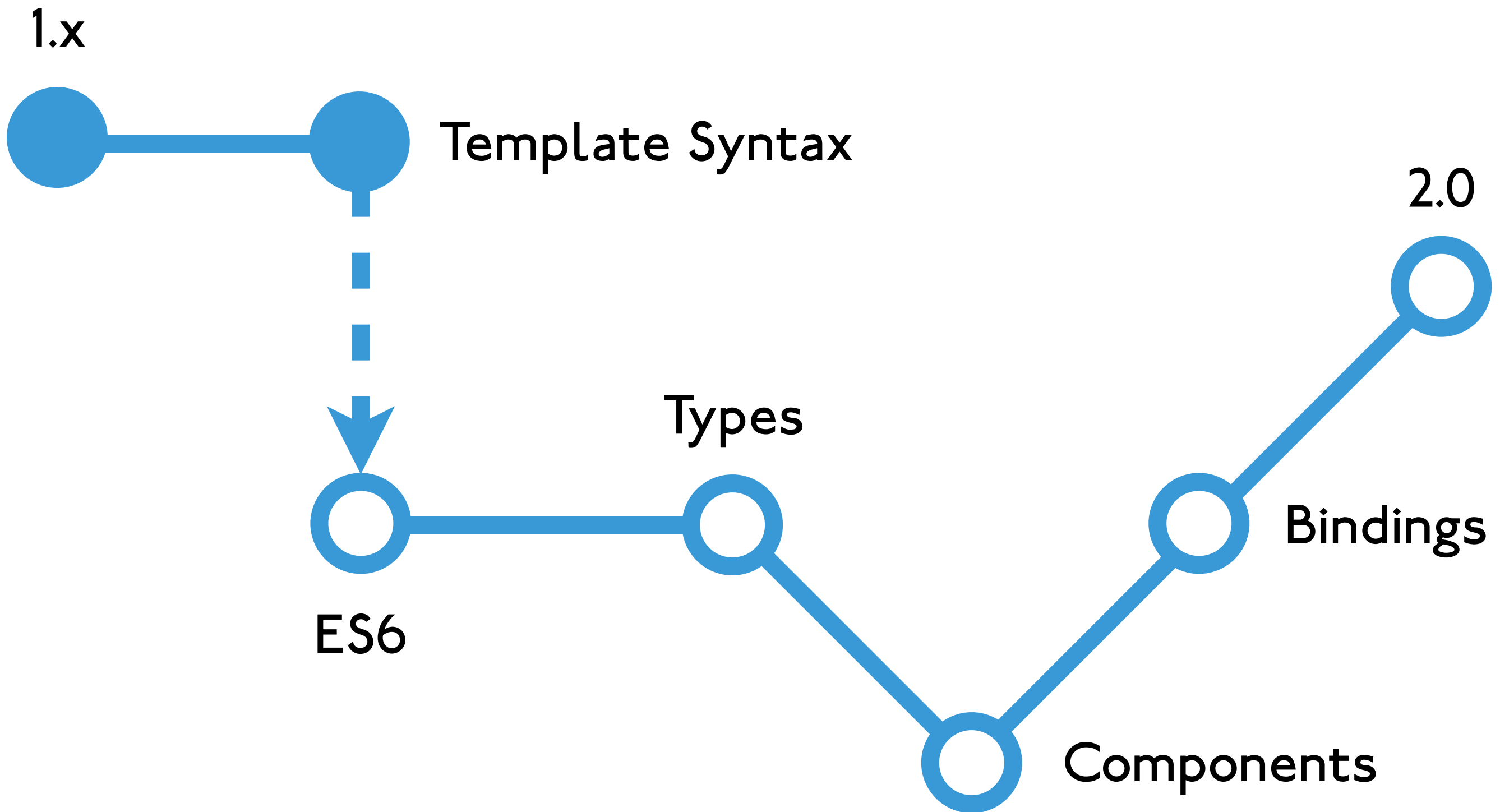
EASIER TO LEARN

*Angular 2.0 templates are closer to
HTML.*

REASONING

*The new syntax makes it easier to
reason about templates.*

THE ROAD



MODULES

ES6 Modules are bundles of 'code' from which you can import specific 'code'.

EXAMPLE

A

```
// MathFuctions.js  
export function square(x) {  
  return x * x;  
}
```

B

```
// main.js  
import {square} from 'MathFuctions';  
  
console.log(square(5)); // logs 25 to console
```

ANGULAR 2.0

```
import {Component, View} from 'angular2/angular2';
```

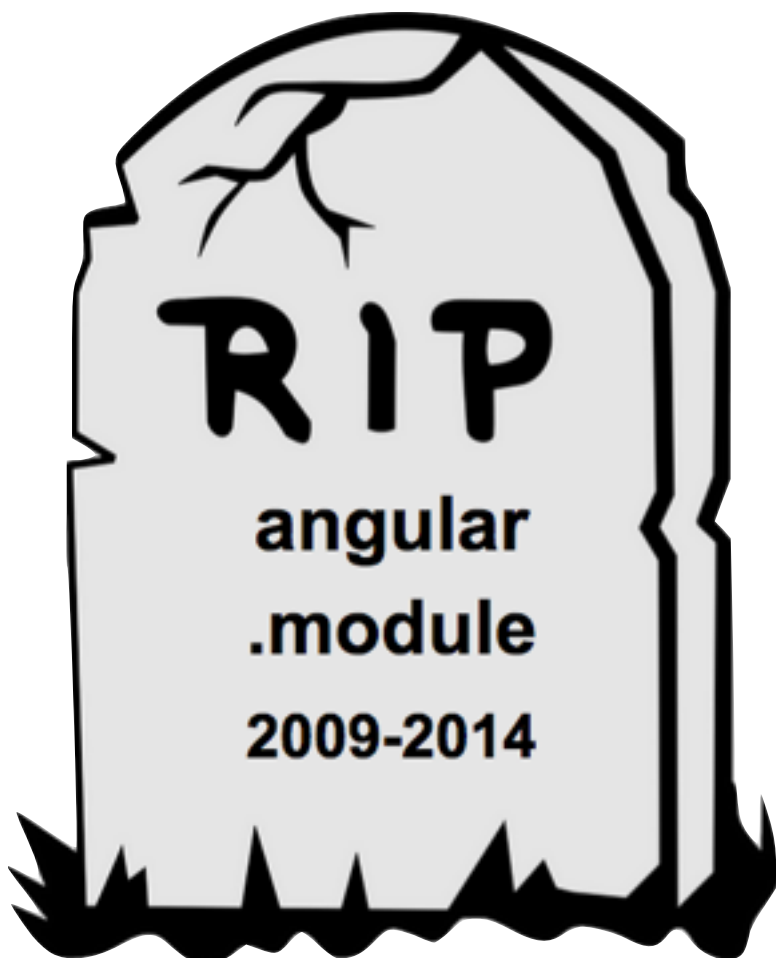
ANGULAR 1.X

```
angular.module('users')  
  .factory('userFactory', ['$http', function($http) {  
    // code for userFactory which uses $http  
  }]);
```

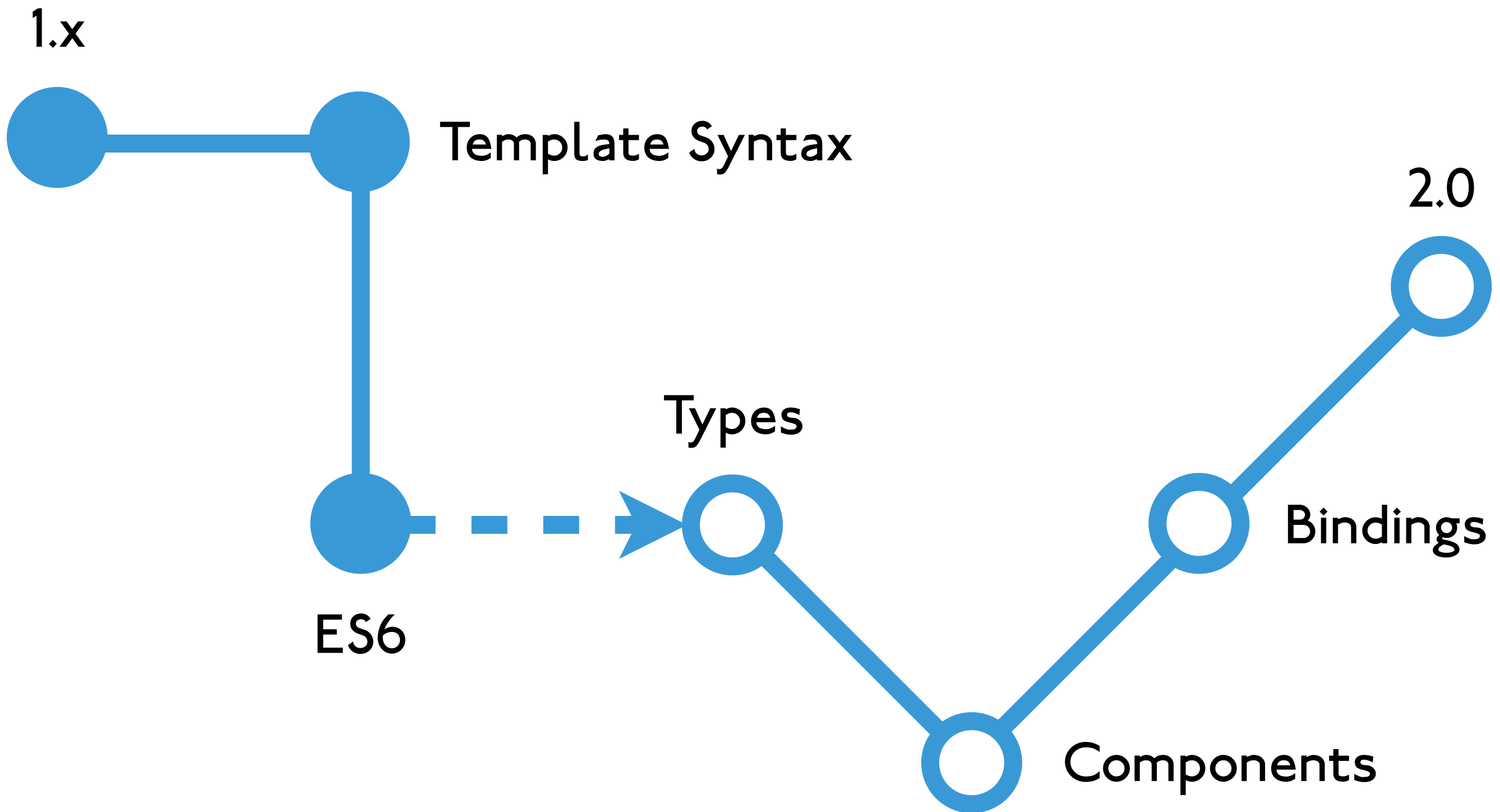

ANGULAR 2.0

*Why create your own a module system,
when the language (ES6) hands it to
you.*

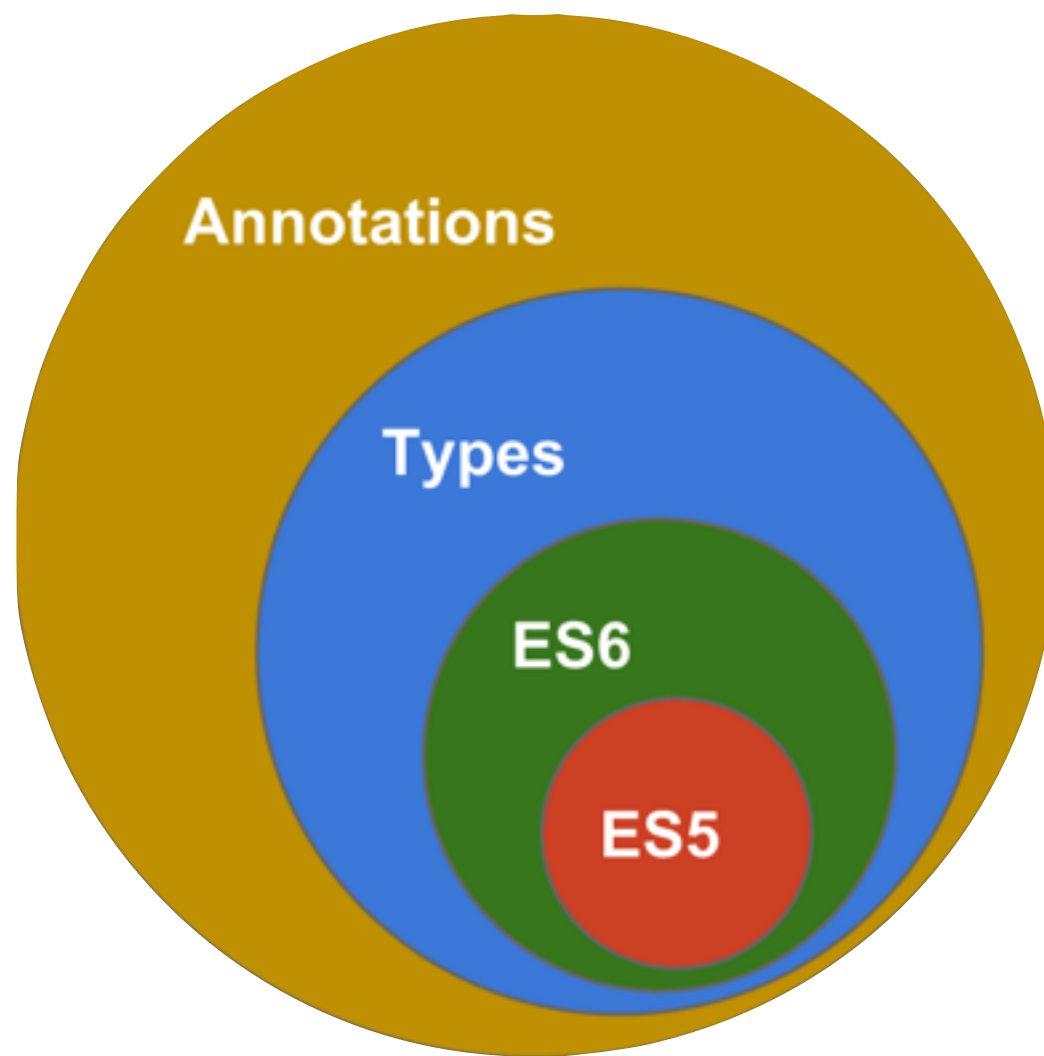
HE'S DEAD JIM



THE ROAD



TYPESCRIPT



ANGULAR 2.0

```
import {Component, View} from 'angular2/angular2';

@Component({selector: 'person'})
@View({
  template: `<p>My name: {{ name }}</p>`
})
class PersonComponent {
  name: string;
  friends: Array<string>;

  constructor(name: string) {
    this.name = name;
  }
}
```

@COMPONENT

```
import {Component, View} from 'angular2/angular2';
```

```
@Component({selector: 'person'})
```

```
@View({
```

```
  template: `<p>My name: {{ name }}</p>`
```

```
})
```

```
class PersonComponent {
```

```
  name: string;
```

```
  friends: Array<string>;
```

```
  constructor(name: string) {
```

```
    this.name = name;
```

```
  }
```

```
}
```



<person></person>

@VIEW

```
import {Component, View} from 'angular2/angular2';

@Component({selector: 'person'})
@View({
  template: `<p>My name: {{ name }}</p>`
})
class PersonComponent {
  name: string;
  friends: Array<string>;

  constructor(name: string) {
    this.name = name;
  }
}
```

TYPES

```
import {Component, View} from 'angular2/angular2';

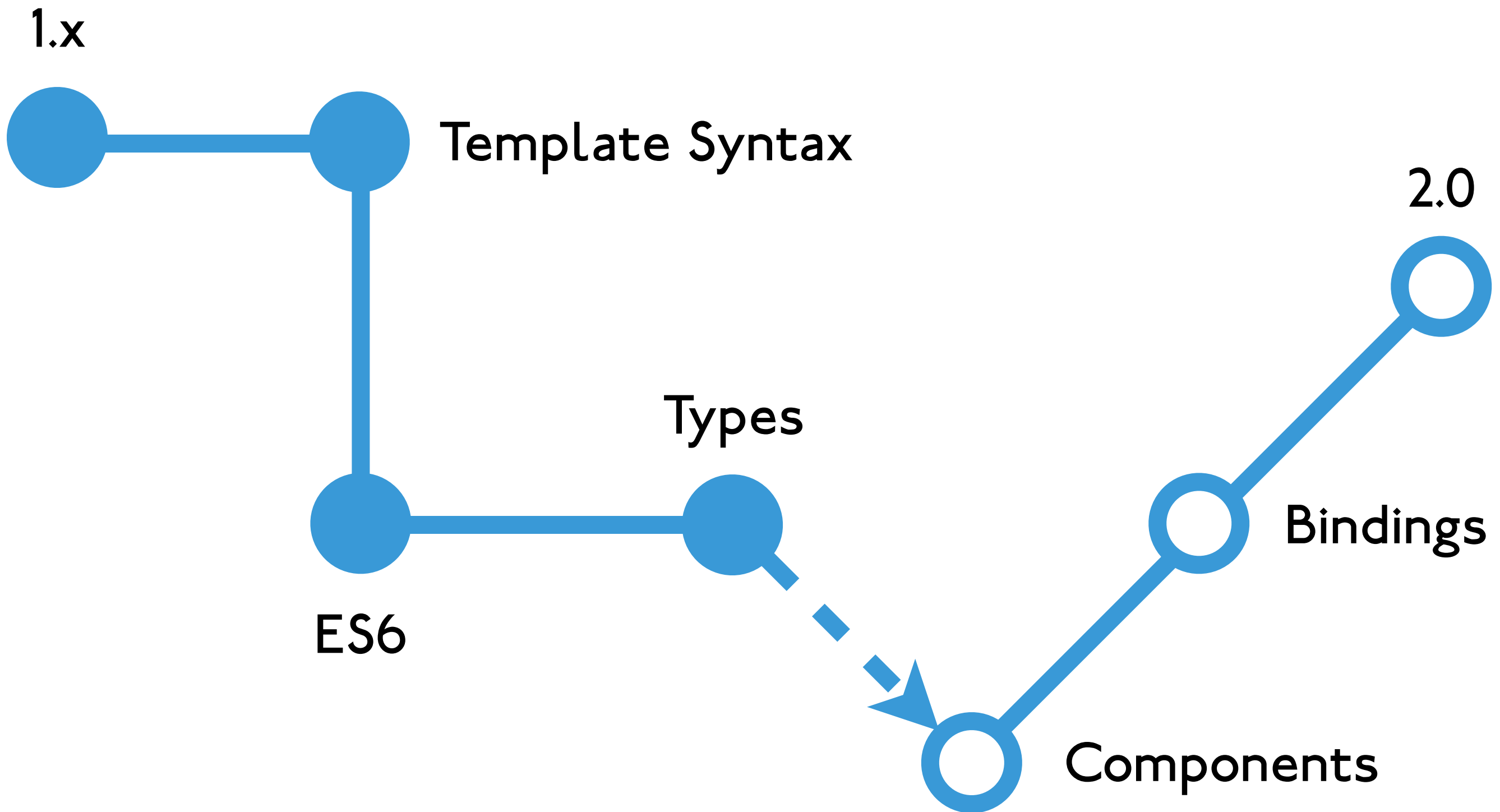
@Component({selector: 'display'})
@View({
  template: `<p>My name: {{ name }}</p>`
})
class PersonComponent {
  name: string;
  friends: Array<string>;

  constructor(name: string) {
    this.name = name;
  }
}
```

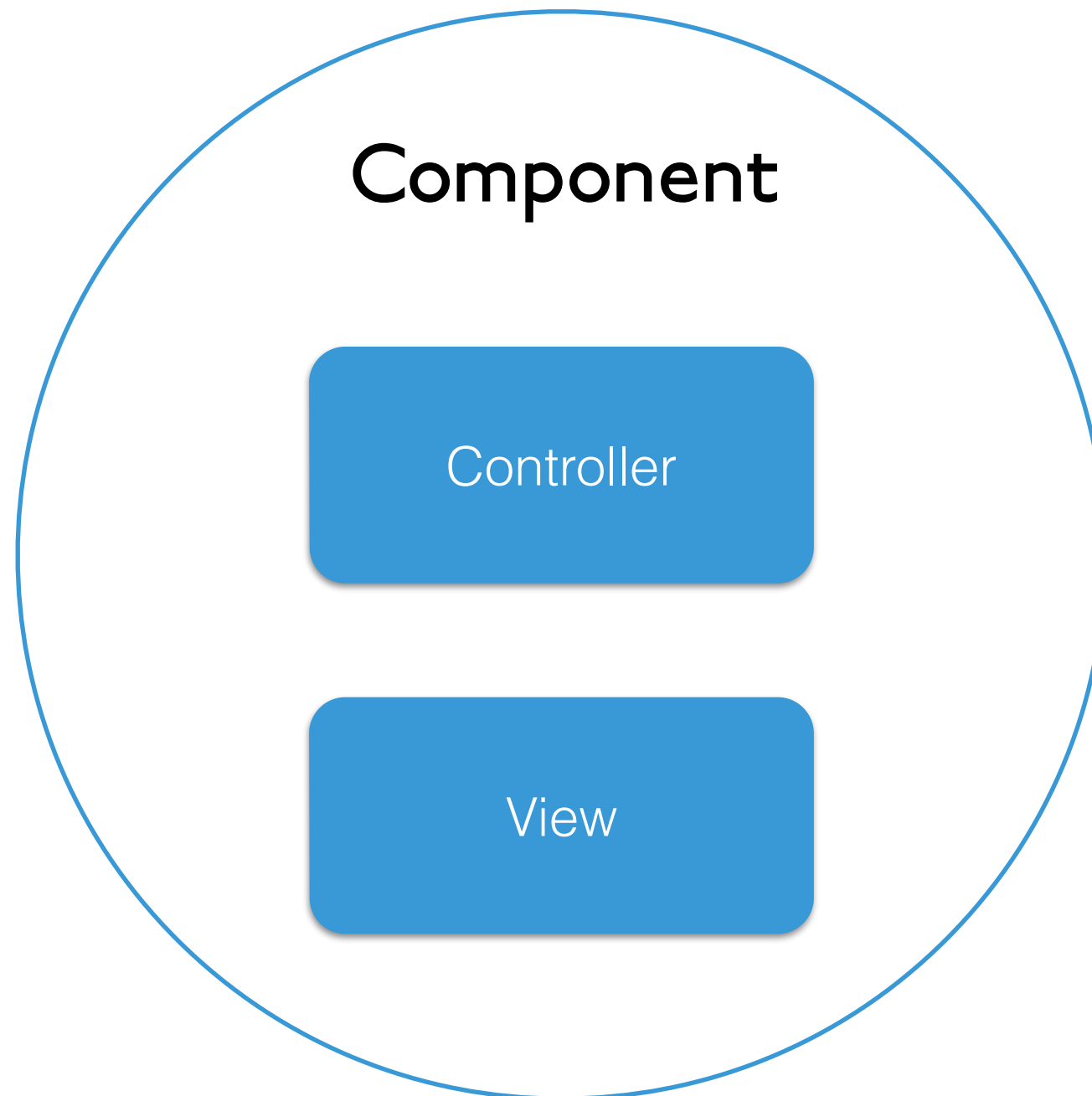

WHY TYPES?

1. IDE's and Text Editors love types.
2. Types help show your intent.

THE ROAD



COMPONENTS

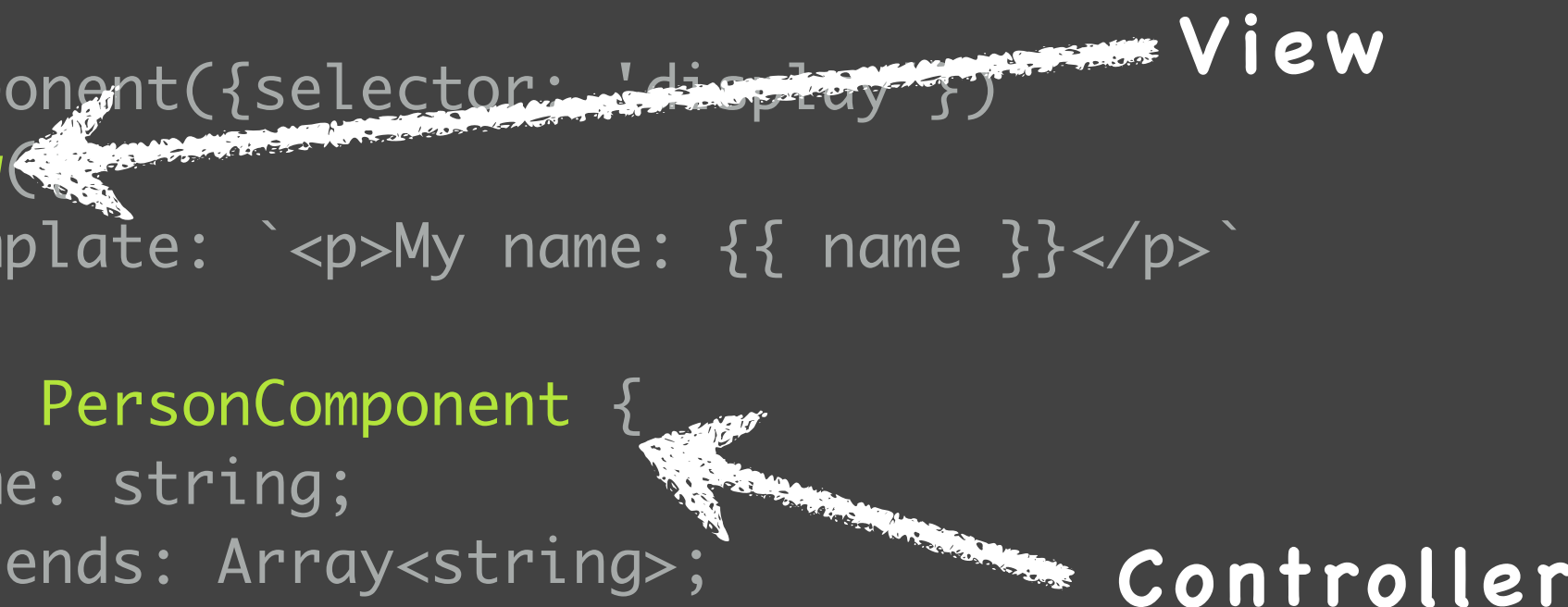


COMPONENT

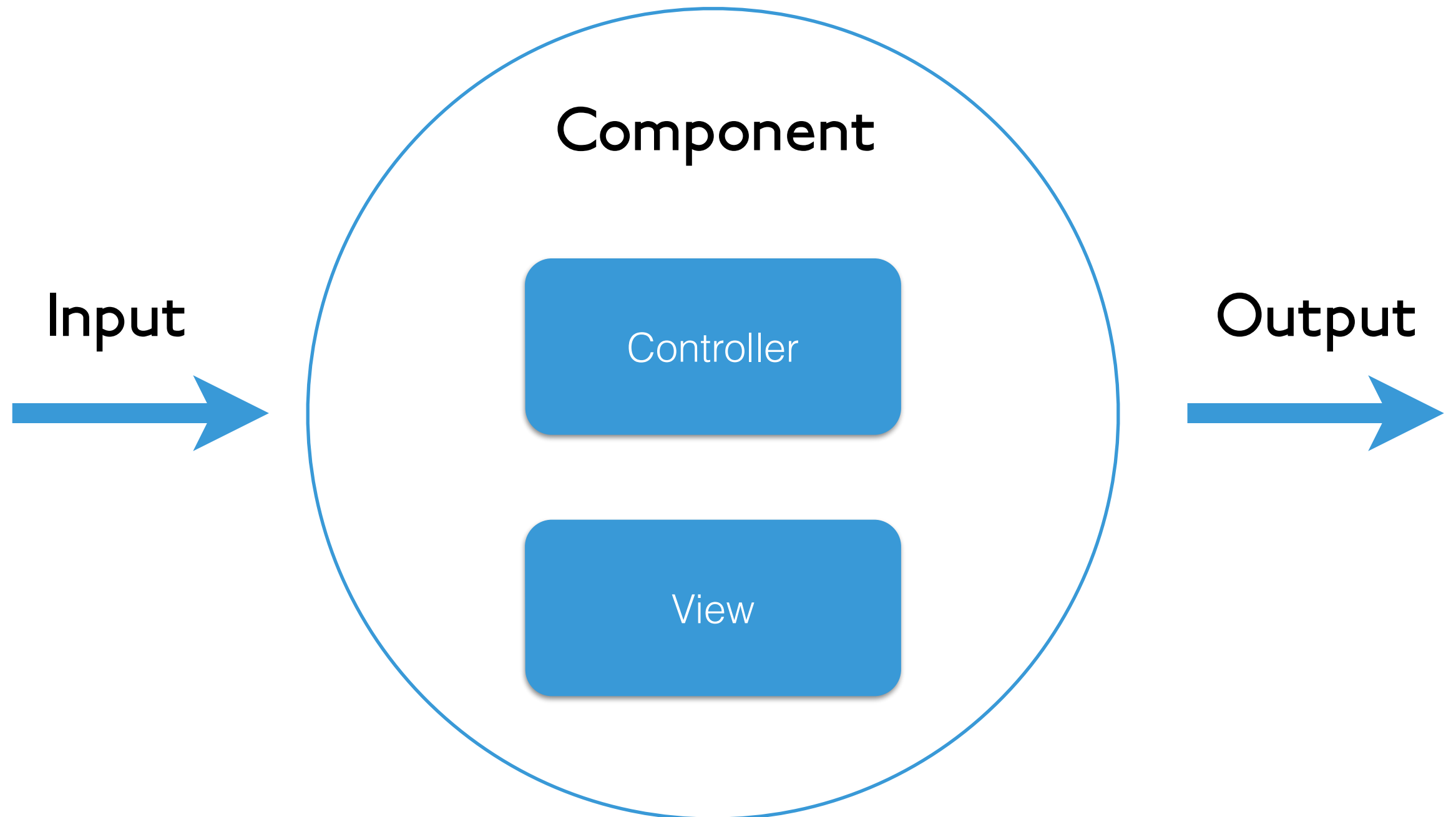
```
import {Component, View} from 'angular2/angular2';  
  
@Component({selector: 'display'})  
@View(  
  template: `<p>My name: {{ name }}</p>`  
)  
class PersonComponent {  
  name: string;  
  friends: Array<string>;  
  
  constructor(name: string) {  
    this.name = name;  
  }  
}
```

View

Controller



INPUT / OUTPUT



I/O



If is explicitly imported

```
import {View, If} from 'angular2/angular2';
```

```
@View({  
  template:  
    `  directives: [If]  
})
```



Components must state
their dependencies (input).

I/O

```
import {Component} from 'angular2/angular2';
```

```
@Component({  
  selector: 'person',  
  properties: { name: 'name' },  
  events: ['upvote']  
})
```



```
<person [name]="firstName"></person>
```

I/O

```
import {Component} from 'angular2/angular2';
```

```
@Component({  
  selector: 'person',  
  properties: { name: 'name' },  
  events: ['upvote']  
})
```

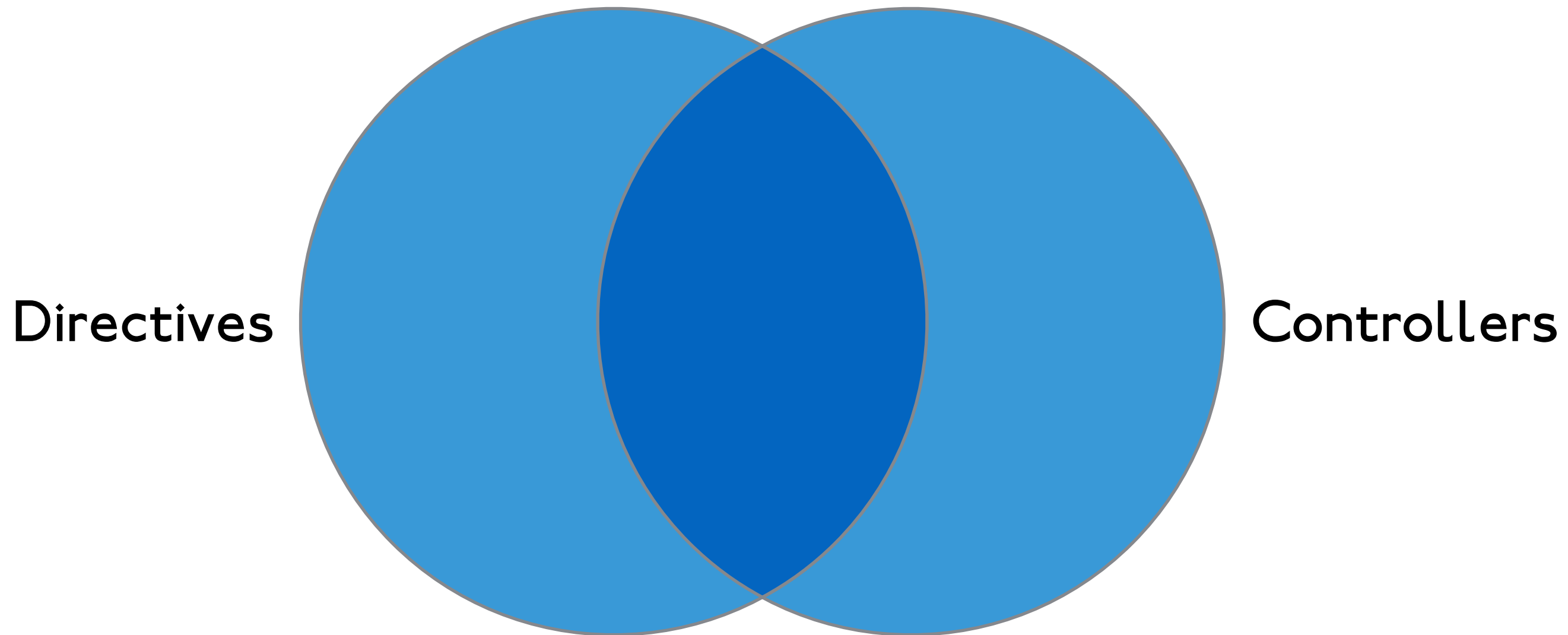


```
<person (upvote)="vote()"></person>
```


WHY COMPONENTS

1. Components have clearly defined inputs and outputs.
2. Components are composable
3. Components can easily be reused.

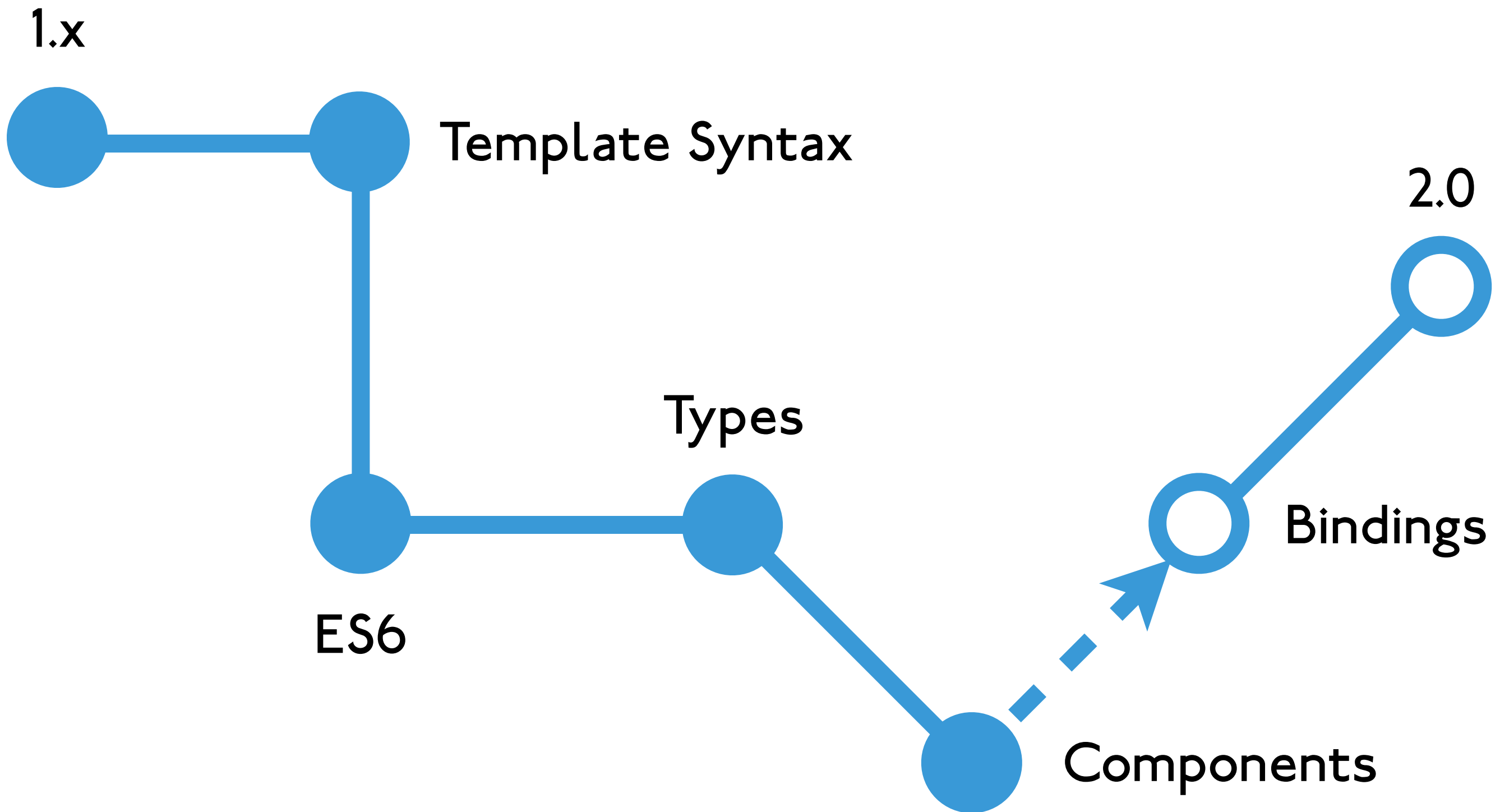
ANGULAR 1.X



THEY LIVE!



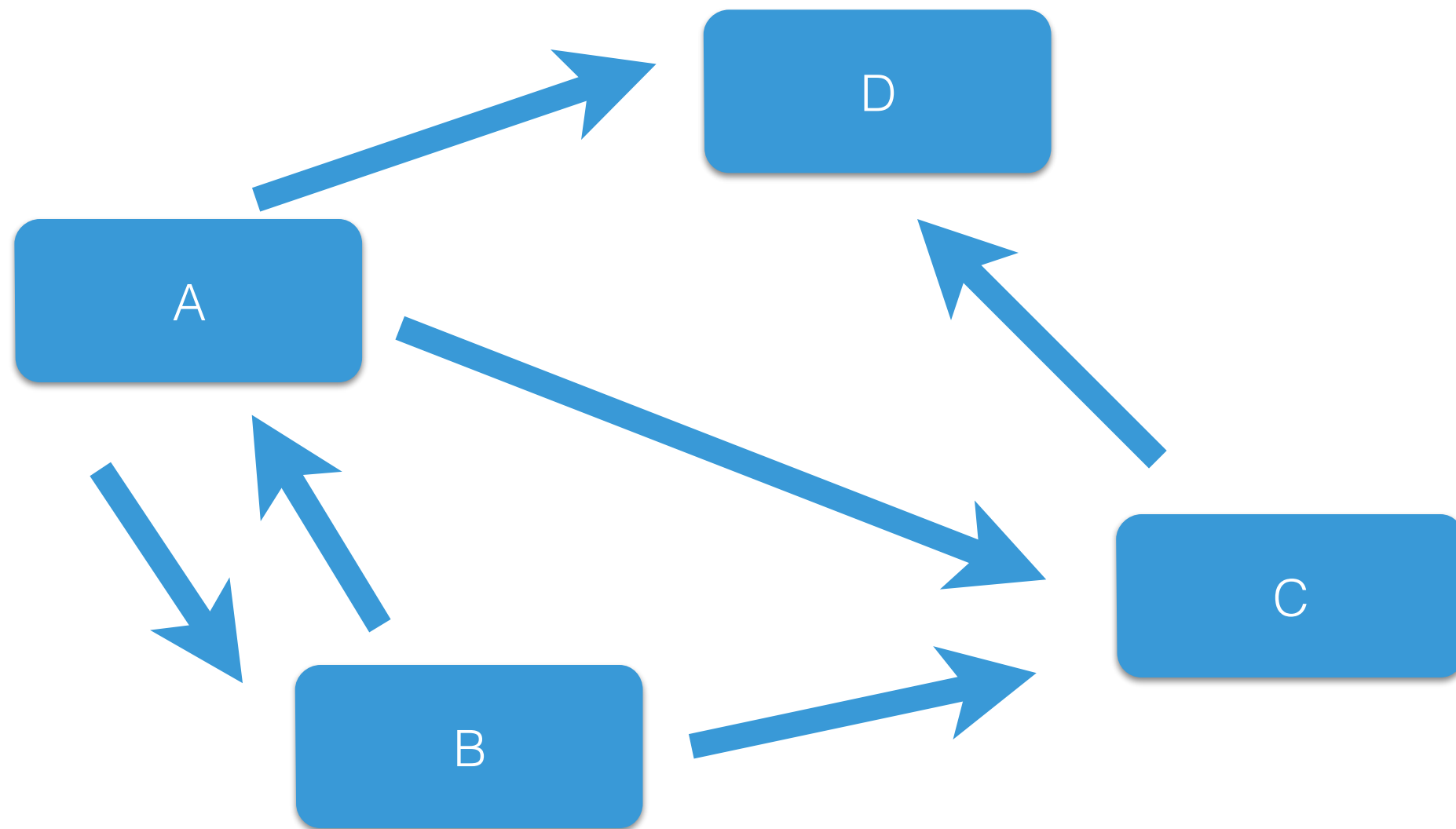
THE ROAD



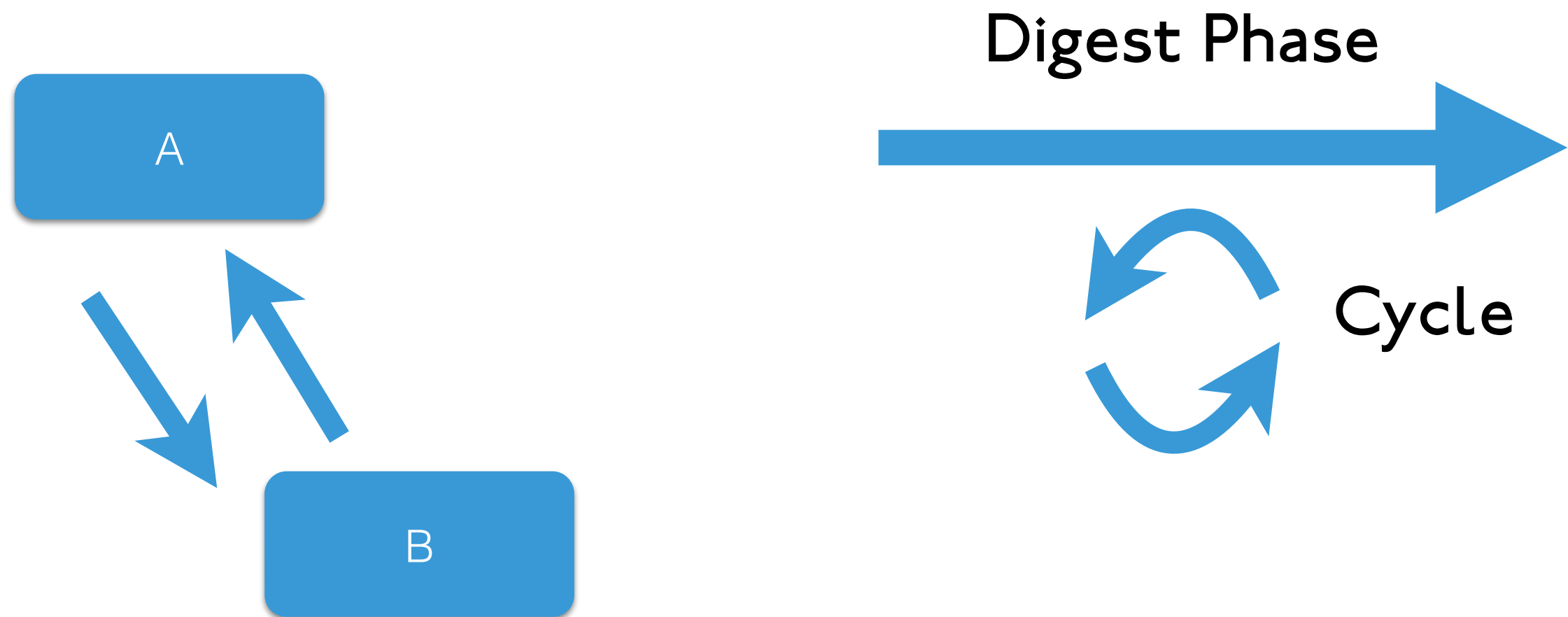
BINDINGS

Angular 2.0 will no longer have two way bindings.

ANGULAR 1.X



EXPENSIVE



UNPREDICTABLE

```
<div ng-controller="PersonController as personController">  
  <h1>Hi {{ personController.person.name }}  
  <person-view person="personController.person"></person-view>  
</div>
```


UNNECESSARY

A

```
<h1>Hi {{ name }}</h1>
```

B

```
<input type="text" value="{{ name }}">
```


ANGULAR 2.0

How does Angular 2.0 handle bindings?

WEATHER APP

Weather

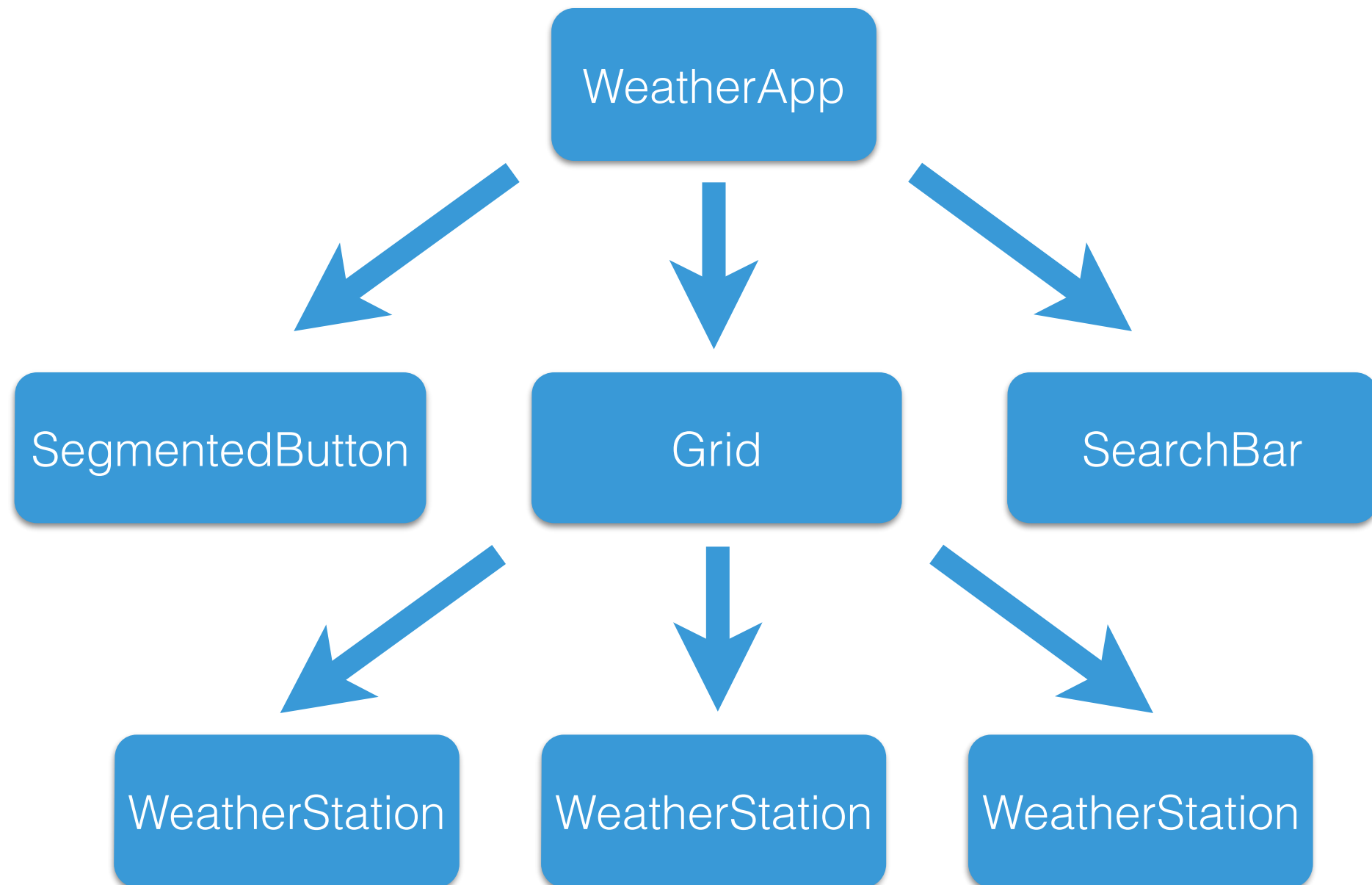
Search for a city

All

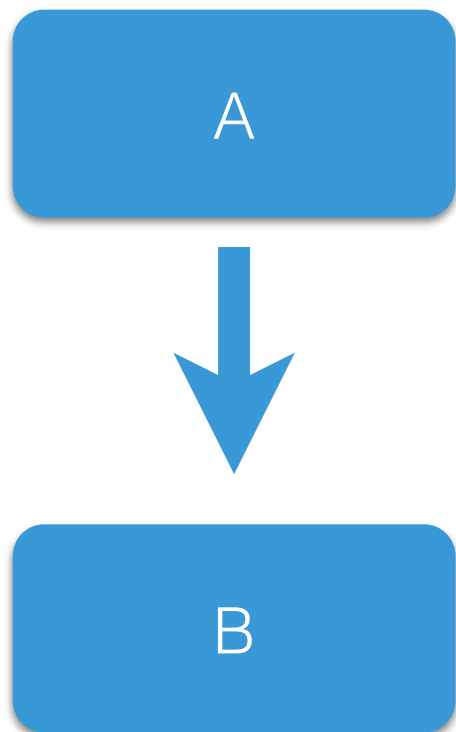
Favorites

 Abu Dhabi 13°C 22%	 Abuja 12°C 56%	 Accra 37°C 20%
 Addis Ababa 3°C 90%	 Algiers -3°C 99%	 Amman 35°C 68%
 Amsterdam 11°C 85%	 Andorra la Vella -5°C 52%	 Ankara 22°C 88%
 Antananarivo 13°C 61%	 Apia 32°C 7%	 Ashgabat 14°C 44%

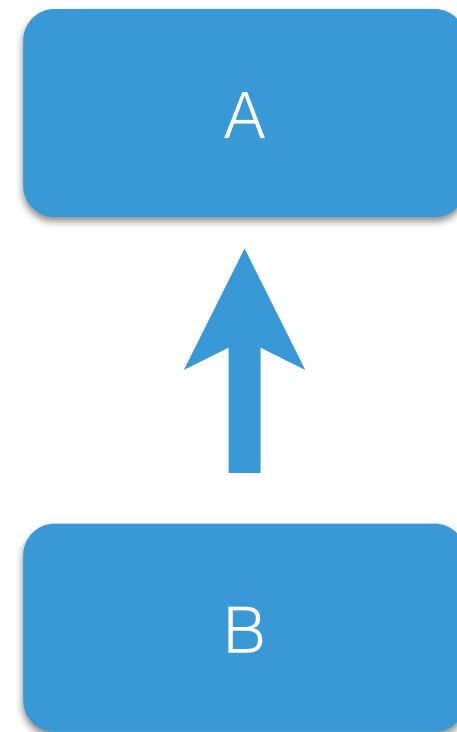
TREE OF COMPONENTS



ANGULAR 2.0



bindings []
go down

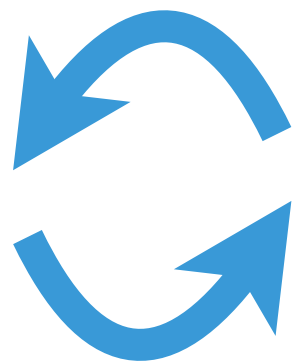


events ()
go up

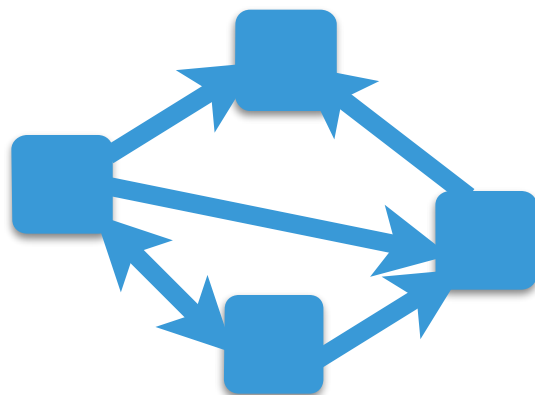
CHANGE DETECTION

1.x

Digest Phase

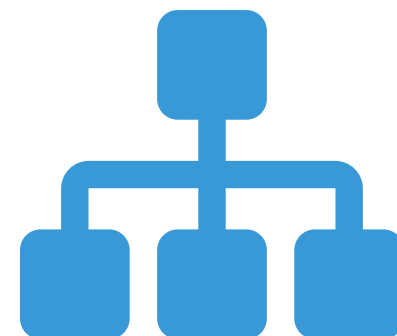


Cycle



2.0

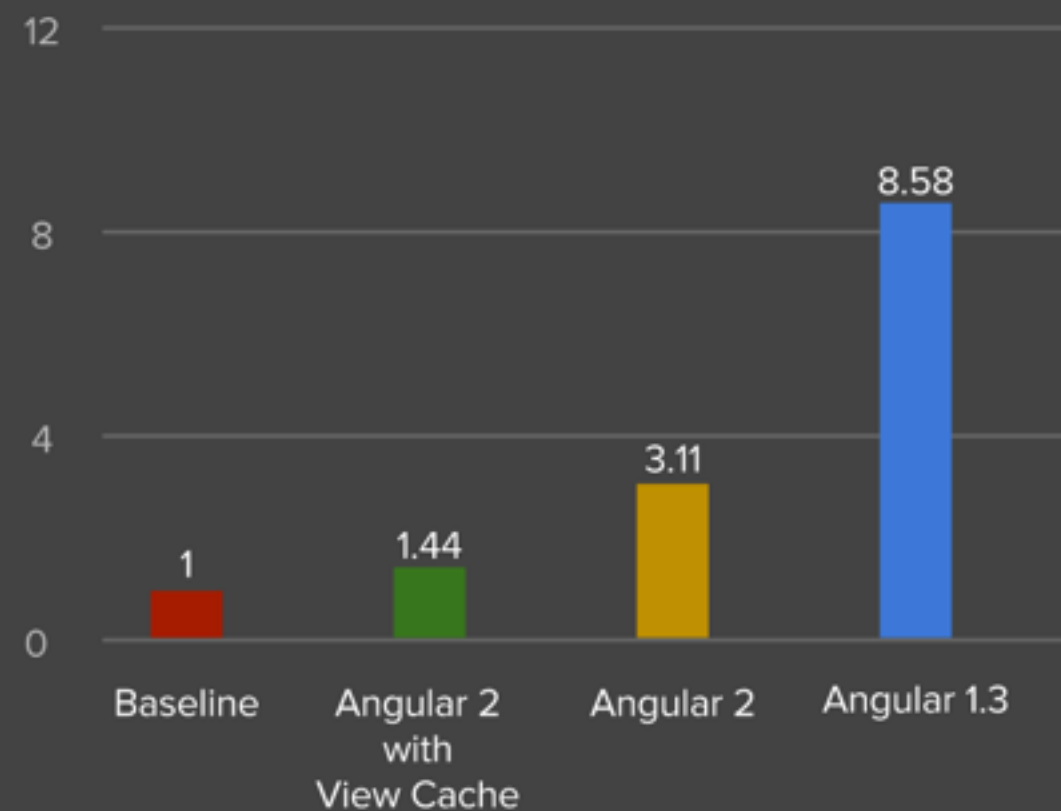
Digest Phase



SPEED

Deep Tree: Speed

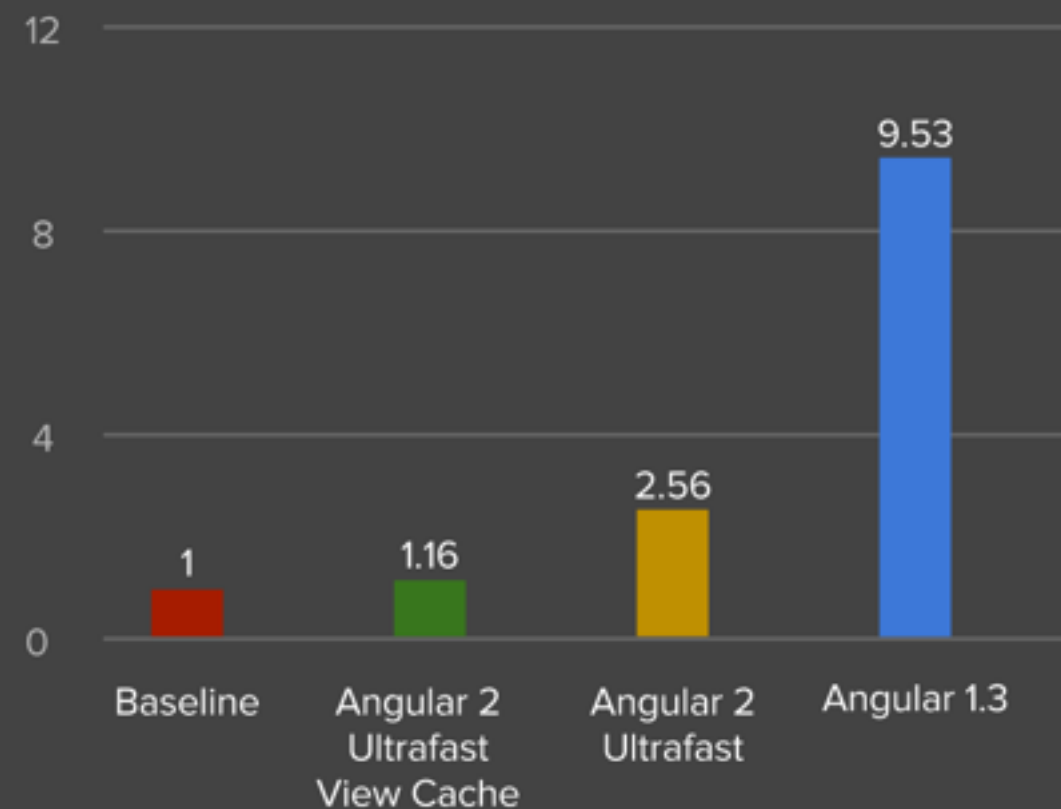
February 9, 2015



MEMORY

Deep Tree: Memory Pressure

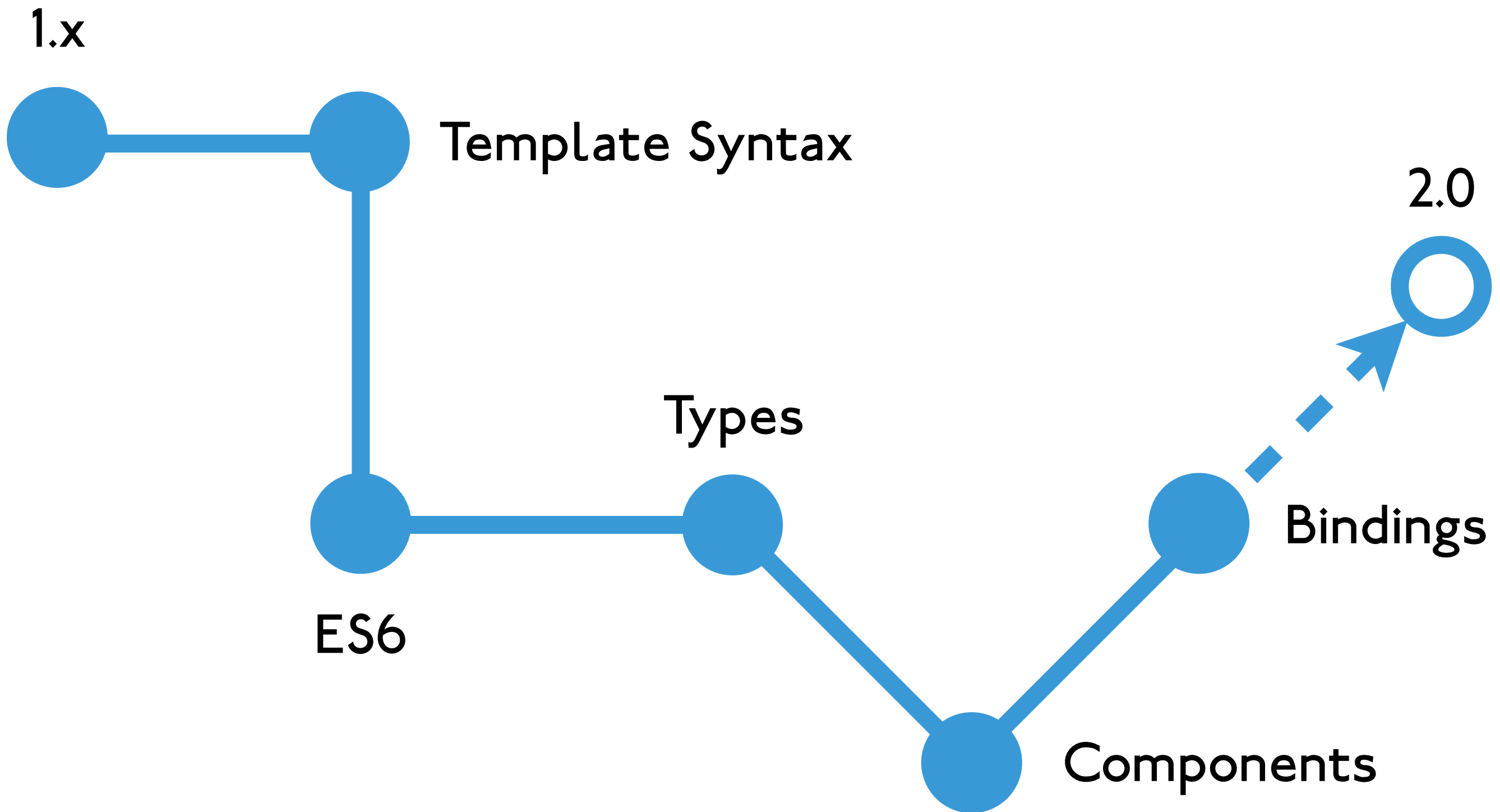
February 9, 2015



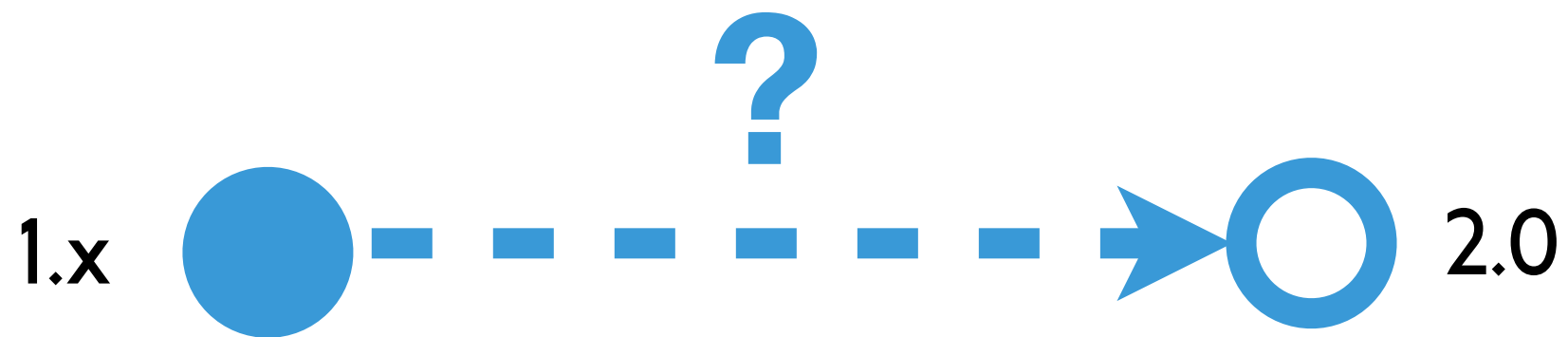
< MEMORY =



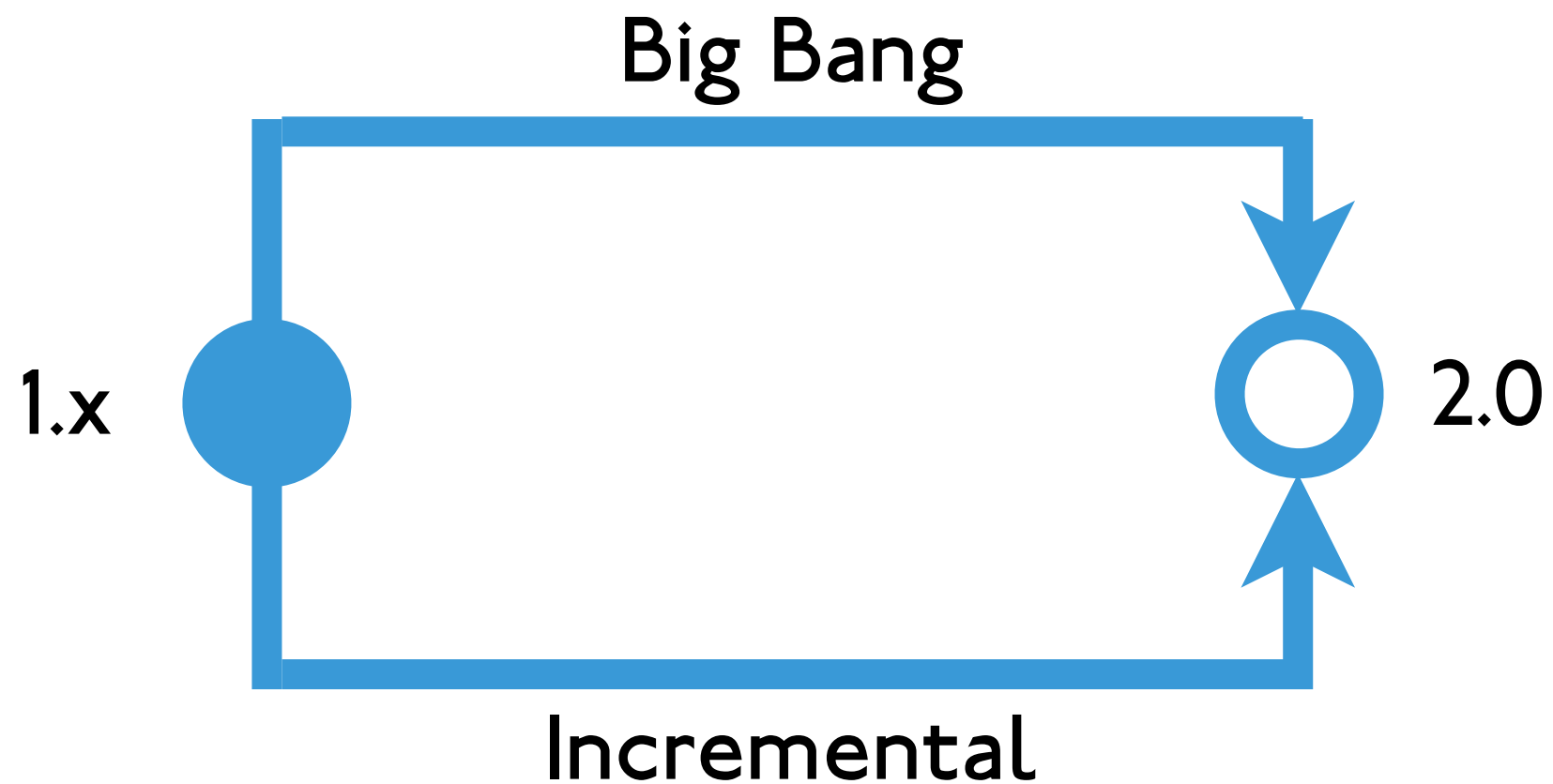
THE ROAD



MIGRATION



MIGRATION



BIG BANG



INCREMENTAL



WHAT TO CHOOSE?

Big Bang	Incremental
Small App	Big App
Few dependencies	Many dependencies
If you have the time	If there is no time

EXAMPLE: A

Big Bang	Incremental
Small App	Big App
Few dependencies	Many dependencies
If you have the time	If there is no time

EXAMPLE: B

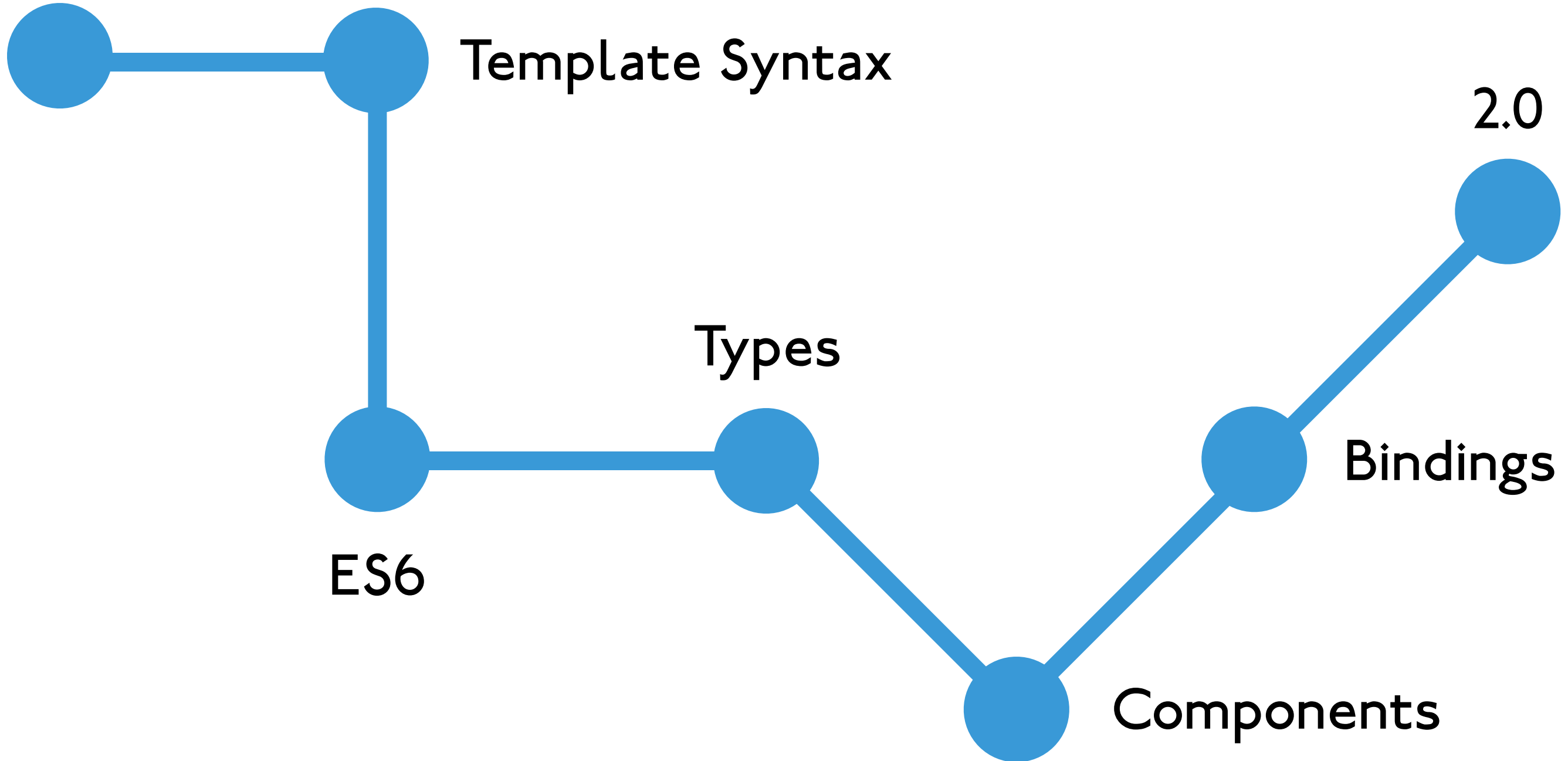
Big Bang	Incremental
Small App	Big App
Few dependencies	Many dependencies
If you have the time	If there is no time

PREPARING FOR 2.0

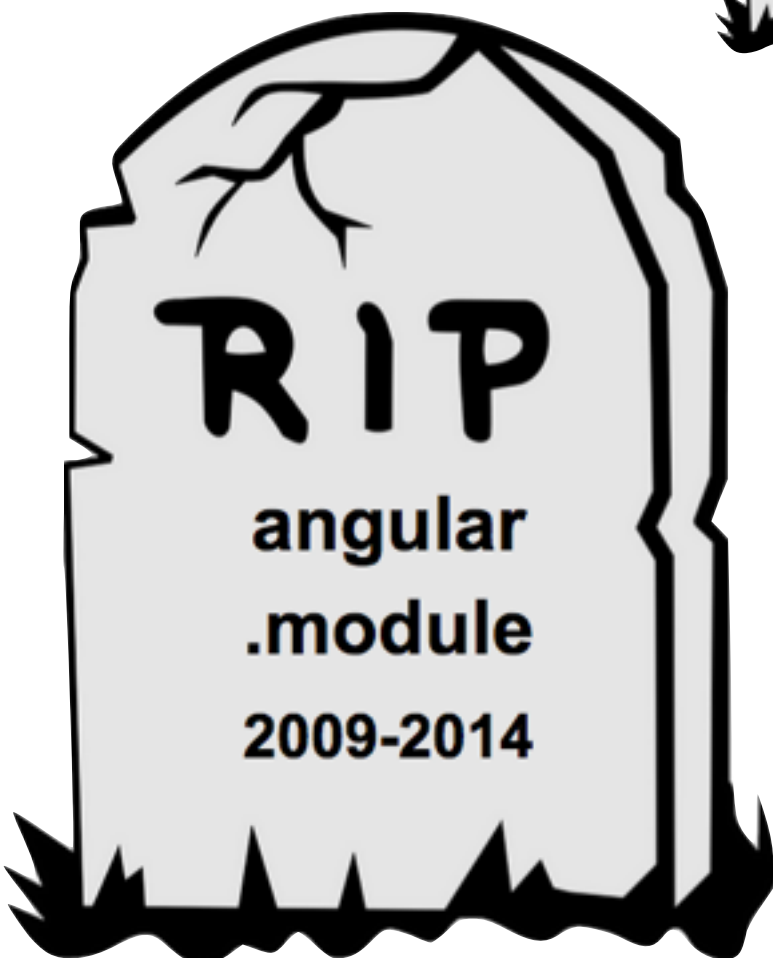
1. Upgrade to 1.4
2. Use the new router
3. Use ES6 classes

DONE

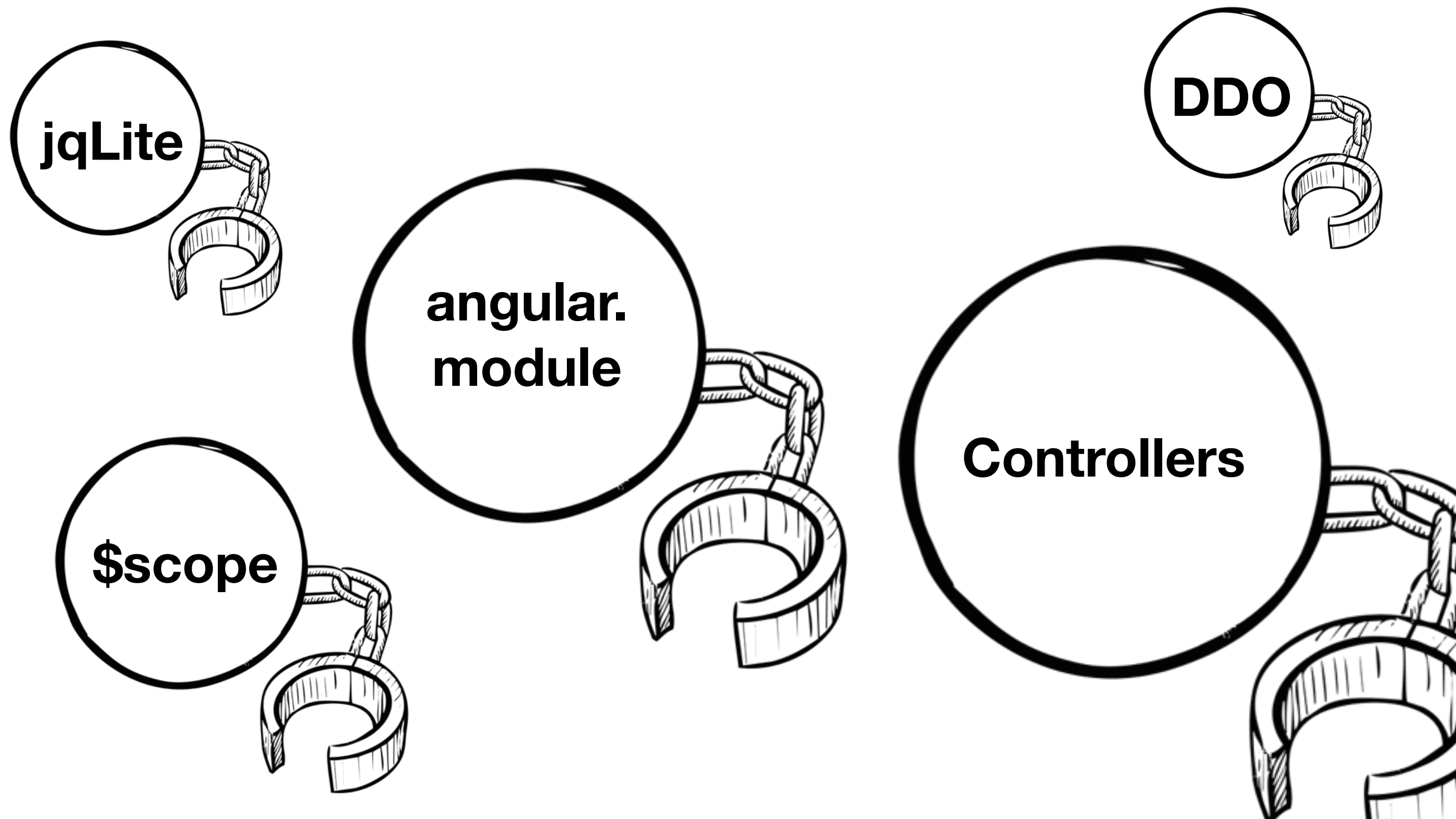
1.x



CONCLUSION



CONCLUSION



THE END

*During the conference you can find me
at the 42's stand. So if you have any
questions you know where to find me.*



Please evaluate
this talk via the
mobile app!

