



# AngularJS

*Jour 2 - Scopes, templates & filters*

# Au programme

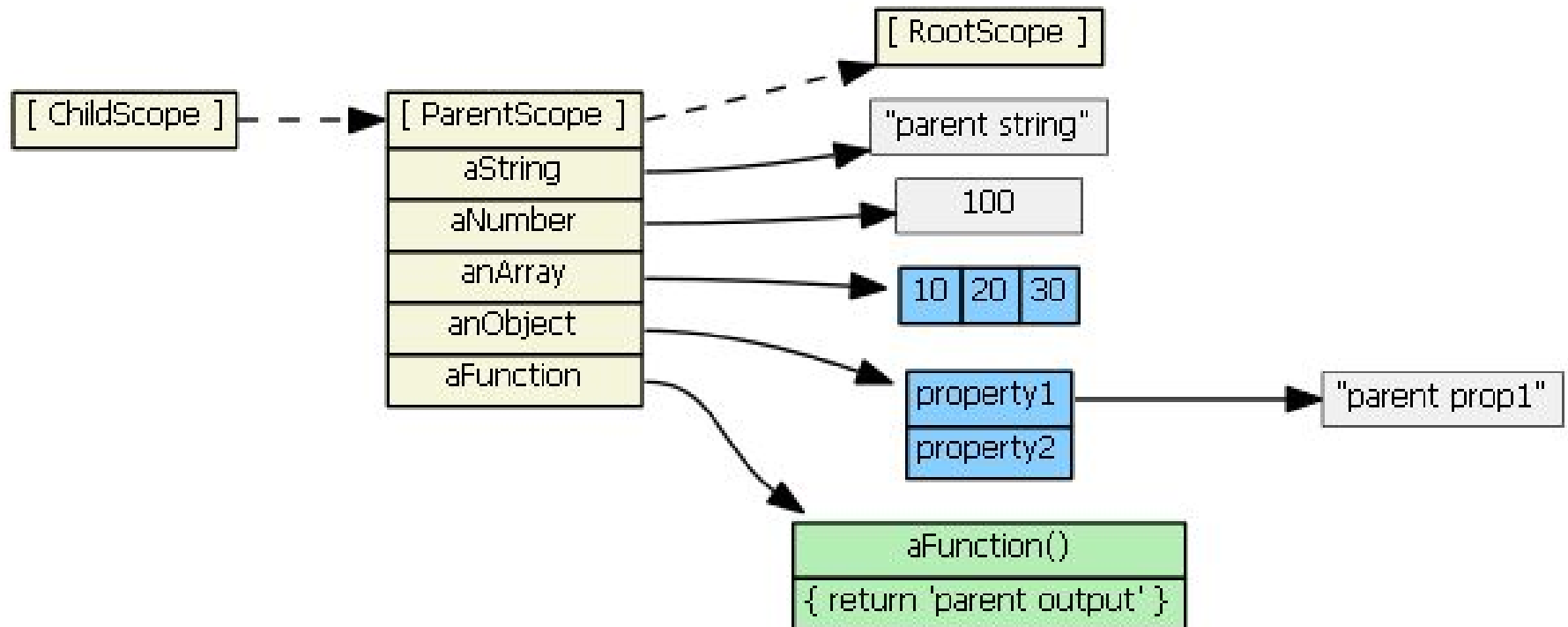
- Le scope
- Le cycle de vie AngularJS
- Les templates
- Les filtres

# 1 - Le scope

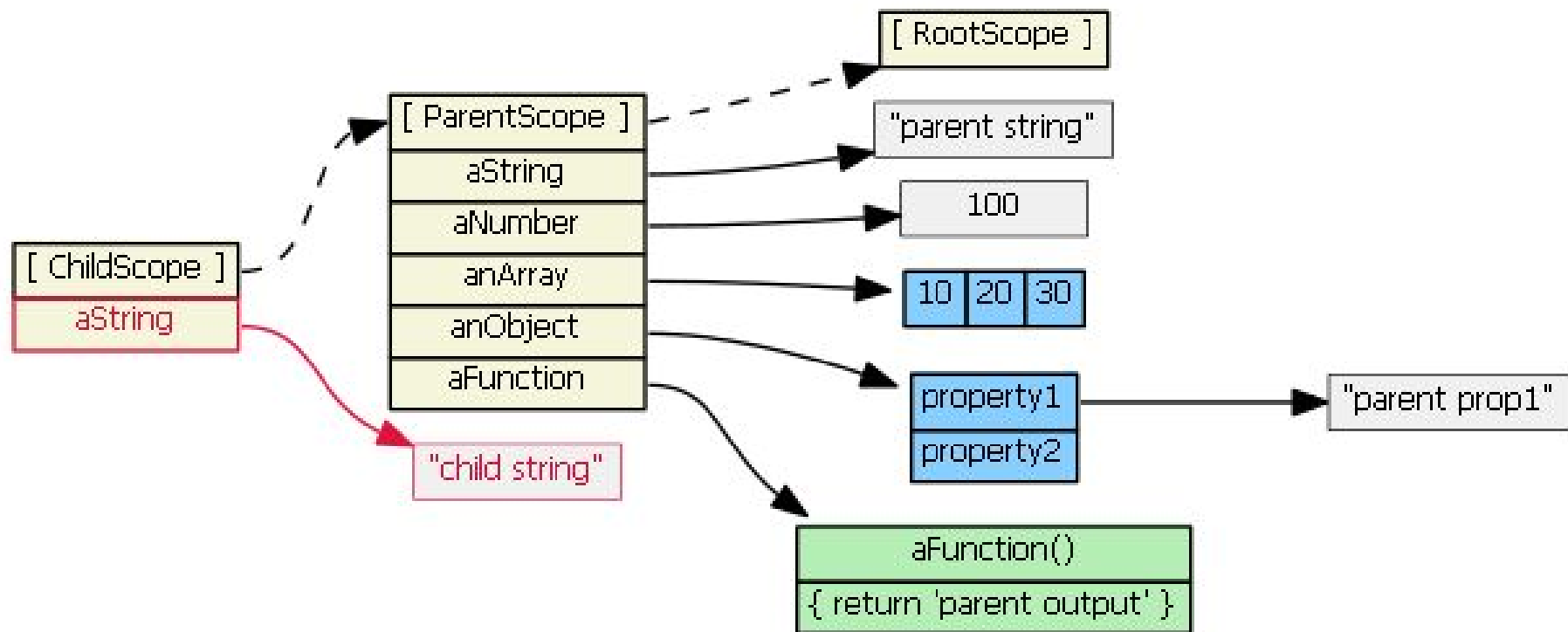
# Le scope

- JavaScript : L'héritage par prototypage

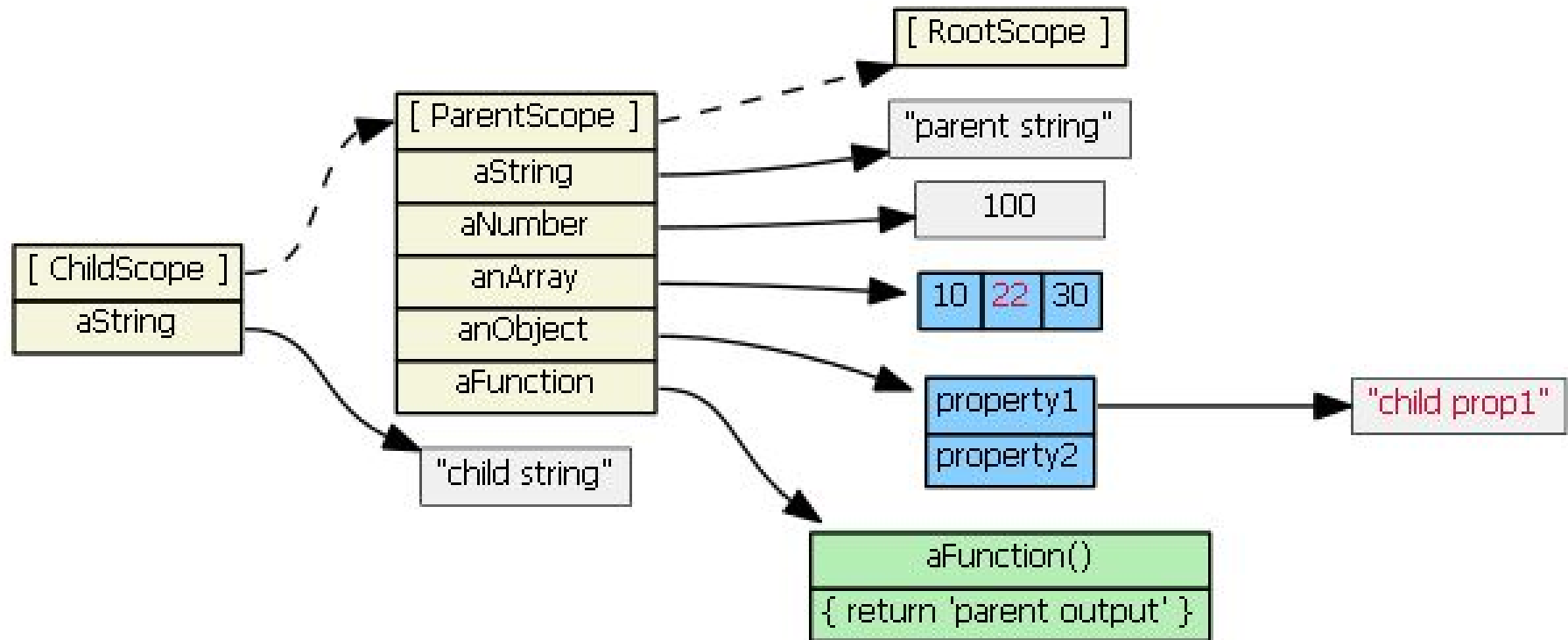
# Le scope - L'héritage par prototypage



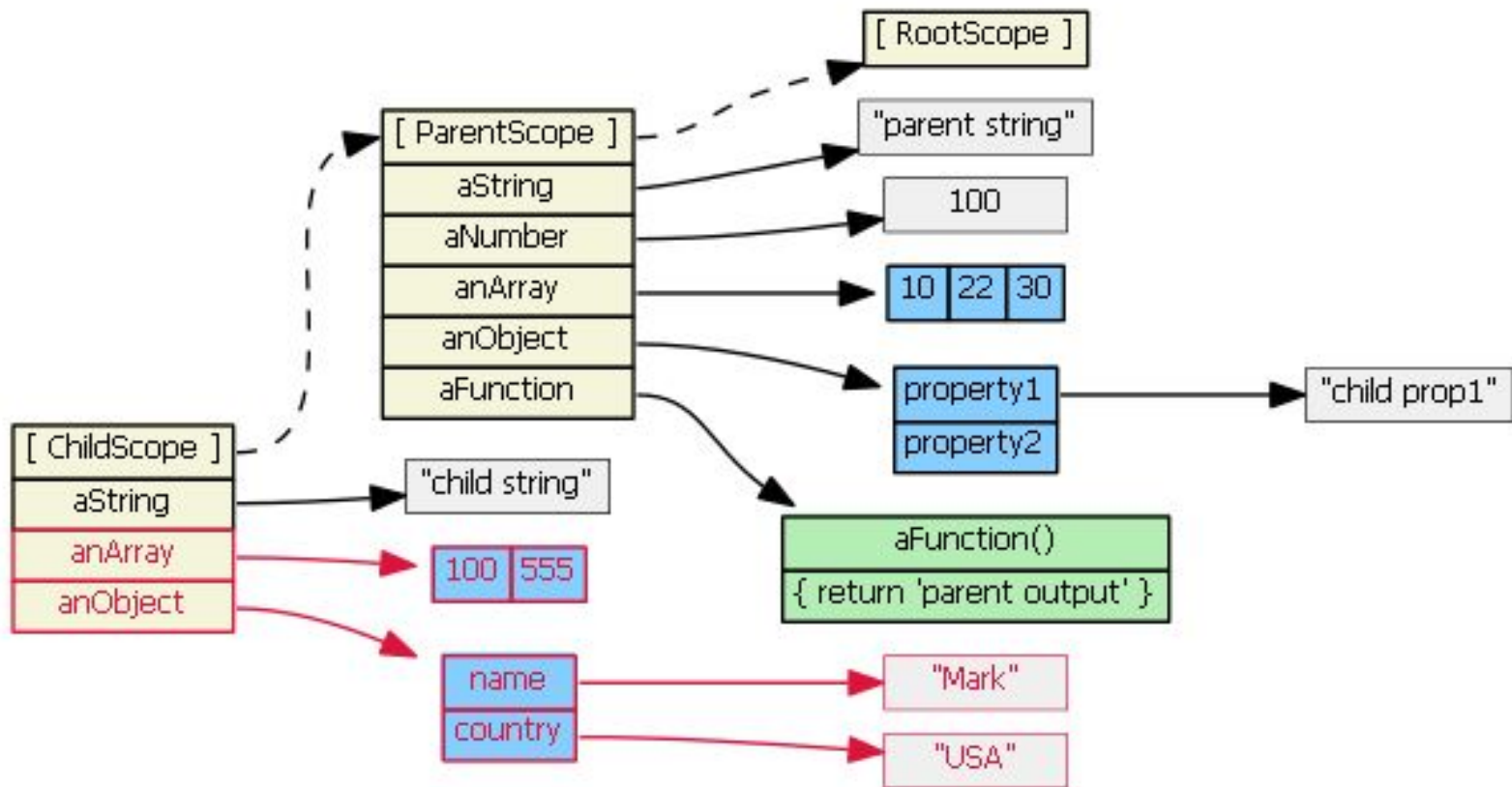
# Le scope - L'héritage par prototypage



# Le scope - L'héritage par prototypage

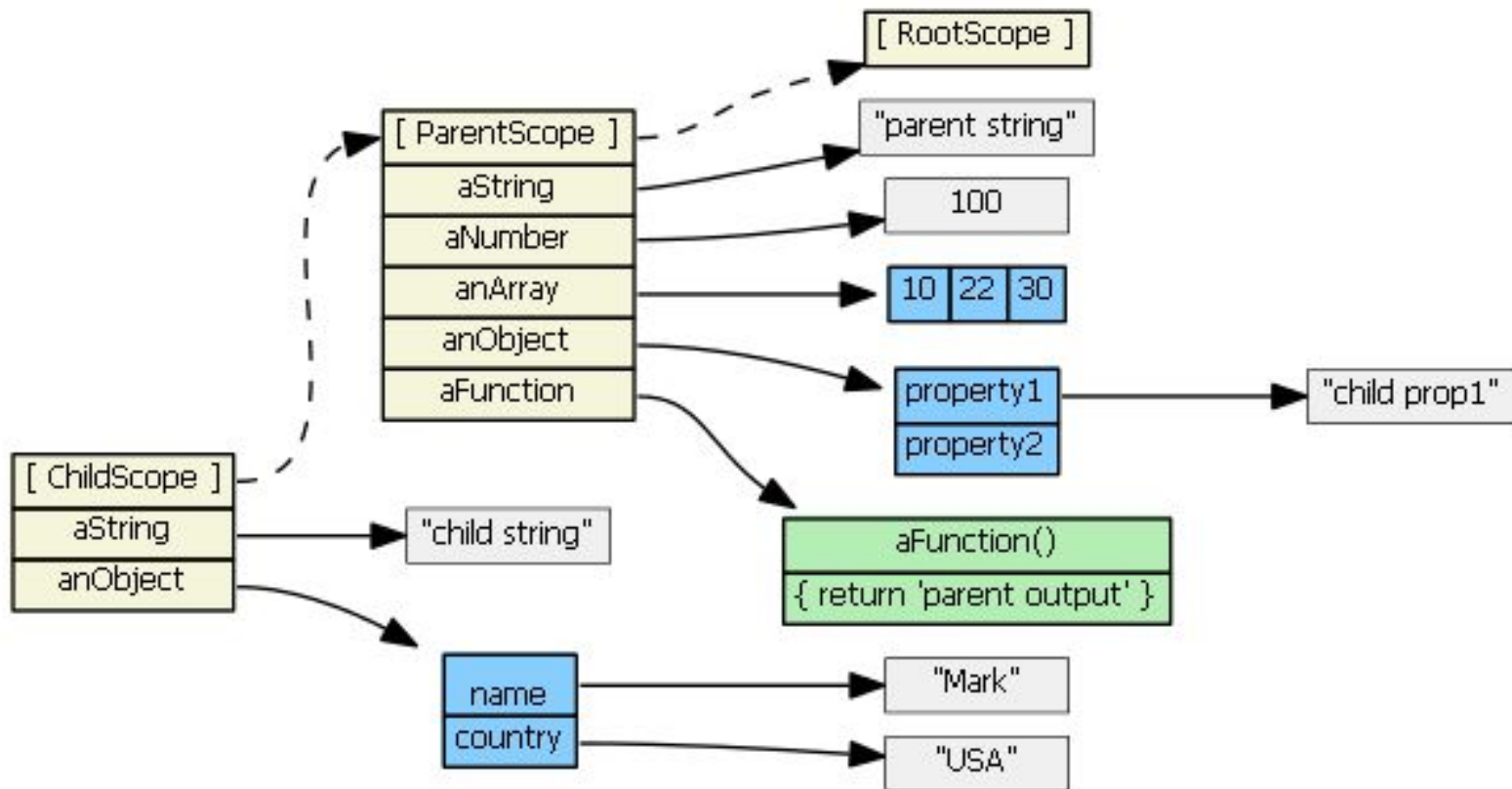


# Le scope - L'héritage par prototypage





# Le scope - L'héritage par prototypage



# Le scope

- JavaScript : L'héritage par prototypage
- L'héritage dans AngularJS

# Le scope - L'héritage dans AngularJS

- Plusieurs types de scope
  - Normal (héritage)
  - Isolé (aucun héritage)
- Toujours un rootScope (ng-app node)
  - \$rootScope

# Le scope

- JavaScript : L'héritage par prototypage
- L'héritage dans AngularJS
- ControllerAs syntax

# Le scope - ControllerAs syntax

- AngularJS 1.1.5
- Permits les nested controllers
- Concrètement
  - `$scope.weather = this;`

# Le scope

- JavaScript : L'héritage par prototypage
- L'héritage dans AngularJS
- ControllerAs syntax
- Attention aux types primitifs

# Le scope - Attention aux types primitifs

- Certaines directives
  - ng-repeat
  - ng-switch
  - ng-include
- Conseil : Toujours travailler sur un objet !

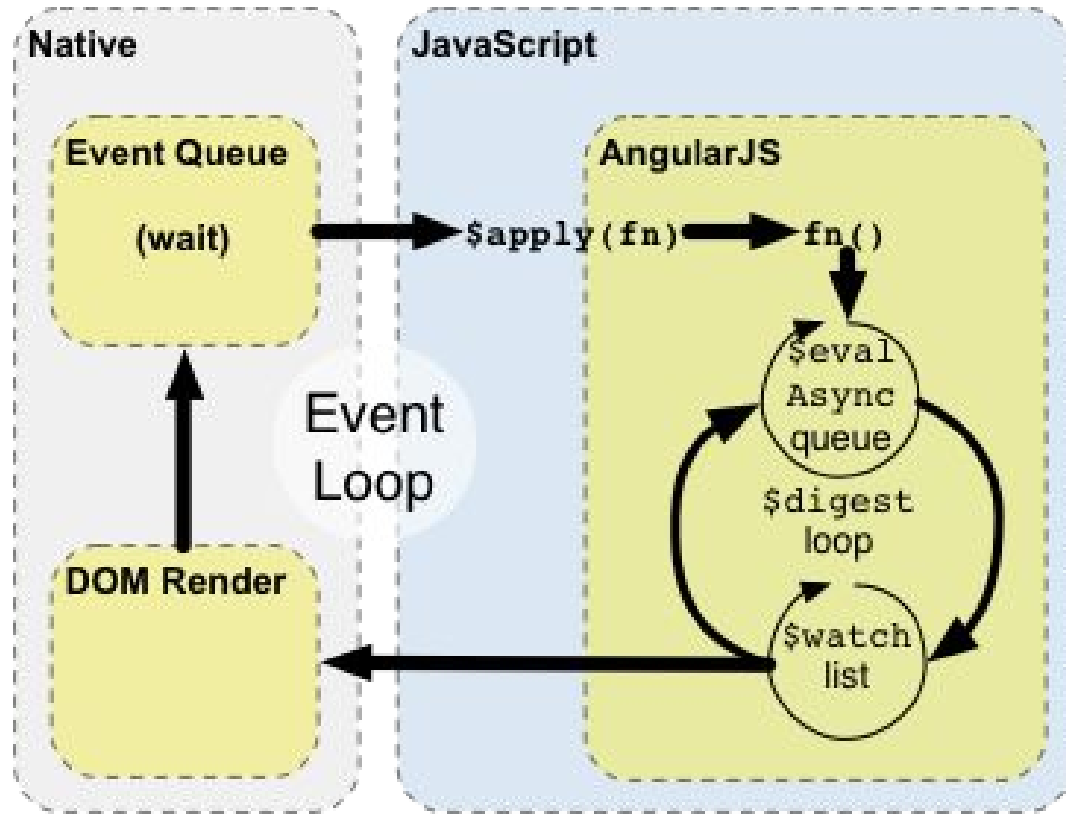
# Le scope

- JavaScript : L'héritage par prototypage
- L'héritage dans AngularJS
- ControllerAs syntax
- Attention aux types primitifs
- DEMO
  - [http://localhost:8080/day\\_02/step\\_01](http://localhost:8080/day_02/step_01)



## 2 - Le cycle de vie

# Le cycle de vie AngularJS



# Le cycle de vie AngularJS

## \$apply et \$digest

- Two-way data binding
- On peut écouter les changements
  - `$scope.$watch`
- DEMO
  - [http://localhost:8080/day\\_02/step\\_02](http://localhost:8080/day_02/step_02)

# Le cycle de vie AngularJS

## Pourquoi & quand appeler \$apply manuellement

- Cycle de vie AngularJS = Monde AngularJS
- Toute opération extérieur
  - Plugin jQuery
  - Requête AJAX (sans les services AngularJS)
  - Code asynchrone
- DEMO
  - [http://localhost:8080/day\\_02/step\\_03](http://localhost:8080/day_02/step_03)

# Le cycle de vie AngularJS

## Performance

- Les humains sont
  - Lent
  - Limité
- Benchmark
  - <http://jsperf.com/angularjs-digest/6>
- L'avenir du dirty-checking
  - Object.Observed dans ECMAScript 7

# 3 - Les templates

# Les templates

## Les directives

- Beaucoup de directives natives
  - <https://docs.angularjs.org/api/ng/directive>
- DEMO
  - [http://localhost:8080/day\\_02/step\\_04](http://localhost:8080/day_02/step_04)

# Les templates

## Les formulaires simples

- HTML5 Validation API
  - <http://www.html5rocks.com/en/tutorials/forms/constraintvalidation/>



# Les templates

## Les formulaires simples

HTML5 Attribute	ng Attribute	Registered Error
required="bool"	ng-required="..."	ngModel.\$error.required
minlength="number"	ng-minlength="number"	ngModel.\$error.minlength
maxlength="number"	ng-maxlength="number"	ngModel.\$error.maxlength
min="number"	ng-min="number"	ngModel.\$error.min
max="number"	ng-max="number"	ngModel.\$error.max
pattern="patternValue"	ng-pattern="patternValue"	ngModel.\$error.pattern

# Les templates

## Les formulaires simples

<code>&lt;input type="..."&gt;</code>	Registered Error
<code>type="email"</code>	<code>ngModel.\$error.email</code>
<code>type="url"</code>	<code>ngModel.\$error.url</code>
<code>type="number"</code>	<code>ngModel.\$error.number</code>
<code>type="date"</code>	<code>ngModel.\$error.date</code>
<code>type="time"</code>	<code>ngModel.\$error.time</code>
<code>type="datetime-local"</code>	<code>ngModel.\$error.datetimelocal</code>
<code>type="week"</code>	<code>ngModel.\$error.week</code>
<code>type="month"</code>	<code>ngModel.\$error.month</code>

# Les templates

## Les formulaires simples

- HTML5 Validation API
  - <http://www.html5rocks.com/en/tutorials/forms/constraintvalidation/>
- DEMO
  - [http://localhost:8080/day\\_02/step\\_05](http://localhost:8080/day_02/step_05)

# 3 - Les filtres

# Les filtres

- Les filtres natifs AngularJS

# Les filtres - Les natifs AngularJS

- <https://docs.angularjs.org/api/ng/filter>
- Des “pure functions”
- `maFunc(a, b) => d`
  - Pour a et b constant, d est identique
- currency, date, filter, json, limitTo, lowercase, number, orderBy, uppercase
- DEMO
  - [http://localhost:8080/day\\_02/step\\_06](http://localhost:8080/day_02/step_06)

# Les filtres

- Les filtres natifs AngularJS
- La création de filtre personnalisé

# Les filtres - La création de filtre

- Syntax identique à tout les composants  
AngularJS
  - `var app = angular.module('myModule', []);`
  - `app.filter('myFilter', function() { return function; });`
- Retourne toujours une fonction
- DEMO
  - [http://localhost:8080/day\\_02/step\\_07](http://localhost:8080/day_02/step_07)



**4 - TP**