AngularJS - Jour 2

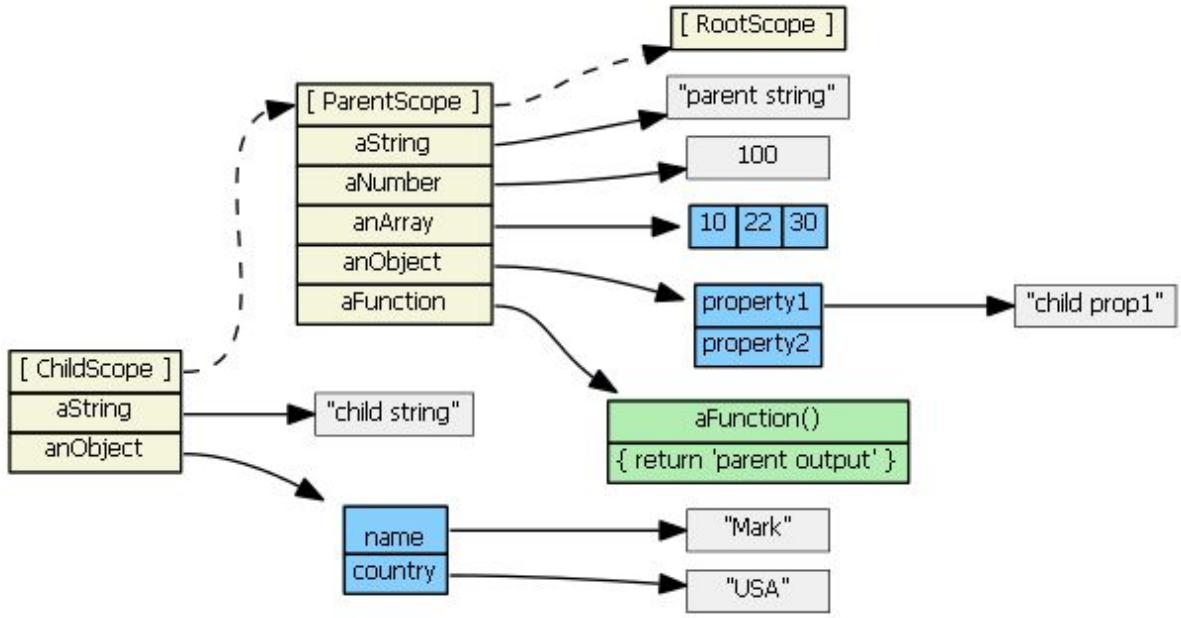
0 - Programme

Au programme:

- Le scope
- Le cycle de vie AngularJS
- Les templates (directive, formulaire)

1 - Le Scope

L'héritage par prototypage dans JavaScript



Ex:

```
childScope.aString === 'parent string'
childScope.anArray[1] === 20
childScope.anObject.property1 === 'parent prop1'
childScope.aFunction() === 'parent output'
```

L'héritage par prototypage dans AngularJS

Un Scope peut être créé de plusieurs manière:

- Normal (héritage: ON)
- Isolé (aucun héritage)

Toujours un rootScope (accessible via le service $rootScope).

ControllerAs syntax

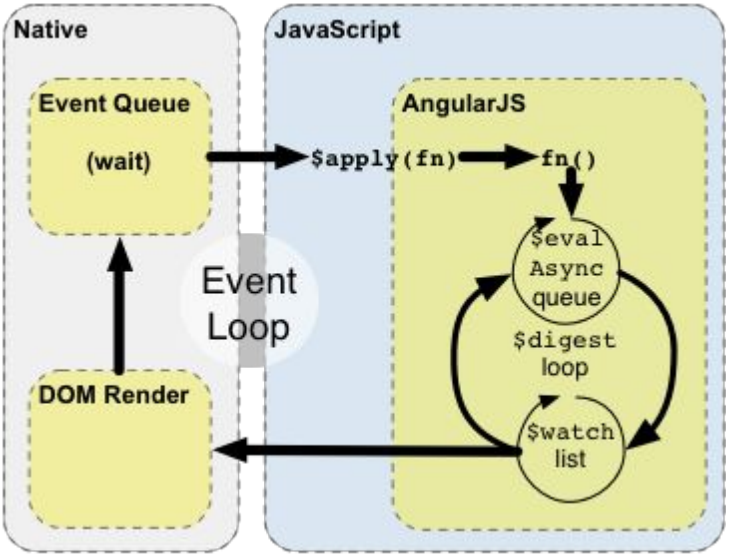
L'intéret de cette syntax est d'éviter l'héritage et de pouvoir accéder à tous les éléments.

Héritage des types primitifs

/!\ ng-include, ng-switch & ng-repeat et l'utilisation des cas primitifs (String, Number)

[DEMO/day_02/step_01]

2 - Le cycle de vie



Dirty-checking => $digest
Update AngularJS Application => $apply

$apply et $digest

AngularJS permet le two-way data binding.
Cela veut dire que l'on peut écouter tout changement sur un élément du Scope avec la méthode $scope.$watch.

[DEMO/day_02/step_02]

Quand appeler $apply manuellement

Cette méthode est à appeler quand on veut utiliser du code Non AngularJS et impacter AngularJS.
Ex: tout code asynchrone setTimeout, request AJAX

[DEMO/day_02/step_03]

Performance

Les humains sont:

- Lent : <50ms est imperceptible et peut être considéeé comme "instantanné"
- Limité : Ne pas afficher plus de 2000 informations sur une même page (mauvaise UI)

Benchmarks (jsperf):
http://jsperf.com/angularjs-digest/6 => 10k watchers

L'avenir du dirty-checking

Object.Observe prévu pour ES7

Lien utile

https://www.youtube.com/watch?v=Mk2WwSxK218

3 - Les templates

Les directives

ng-if, ng-show, ng-class, ng-src, ng-click, ng-copy, ng-paste, ng-href, ng-switch, etc...
[DEMO/day_02/step_04]

Les formulaires

HTML5 Validation API

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

<input type="...">	Registered Error
type="email"	ngModel.$error.email
type="url"	ngModel.$error.url
type="number"	ngModel.$error.number
type="date"	ngModel.$error.date
type="time"	ngModel.$error.time
type="datetime-local"	ngModel.$error.datetimelocal
type="week"	ngModel.$error.week
type="month"	ngModel.$error.month

[DEMO/day_02/step_05]

4 - Les filtres

Les filtres natifs AngularJS

Les filtres sont des “pure functions” qui permette de muter une variable passée en paramètre.
Beaucoup de filtre fournis avec AngularJS : currency, date, filter, json, limitTo, lowercase, number, orderBy, uppercase

[DEMO/day_02/step_06]

Création de filtre personnalisé

On crée un filter comme on crée un controlleur.
app.filter(‘NameFilter’, function() {});
Un filtre est une fonction, qui prend x paramètre en entré et retourne une valeur.
[DEMO/day_02/step_07]

5 - TP

- Créer un nouveau dossier
- Initialiser Bower (bower.json)
- Installer angular
- Créer page qui liste des utilisateurs
  {firstname: string, lastname: string}