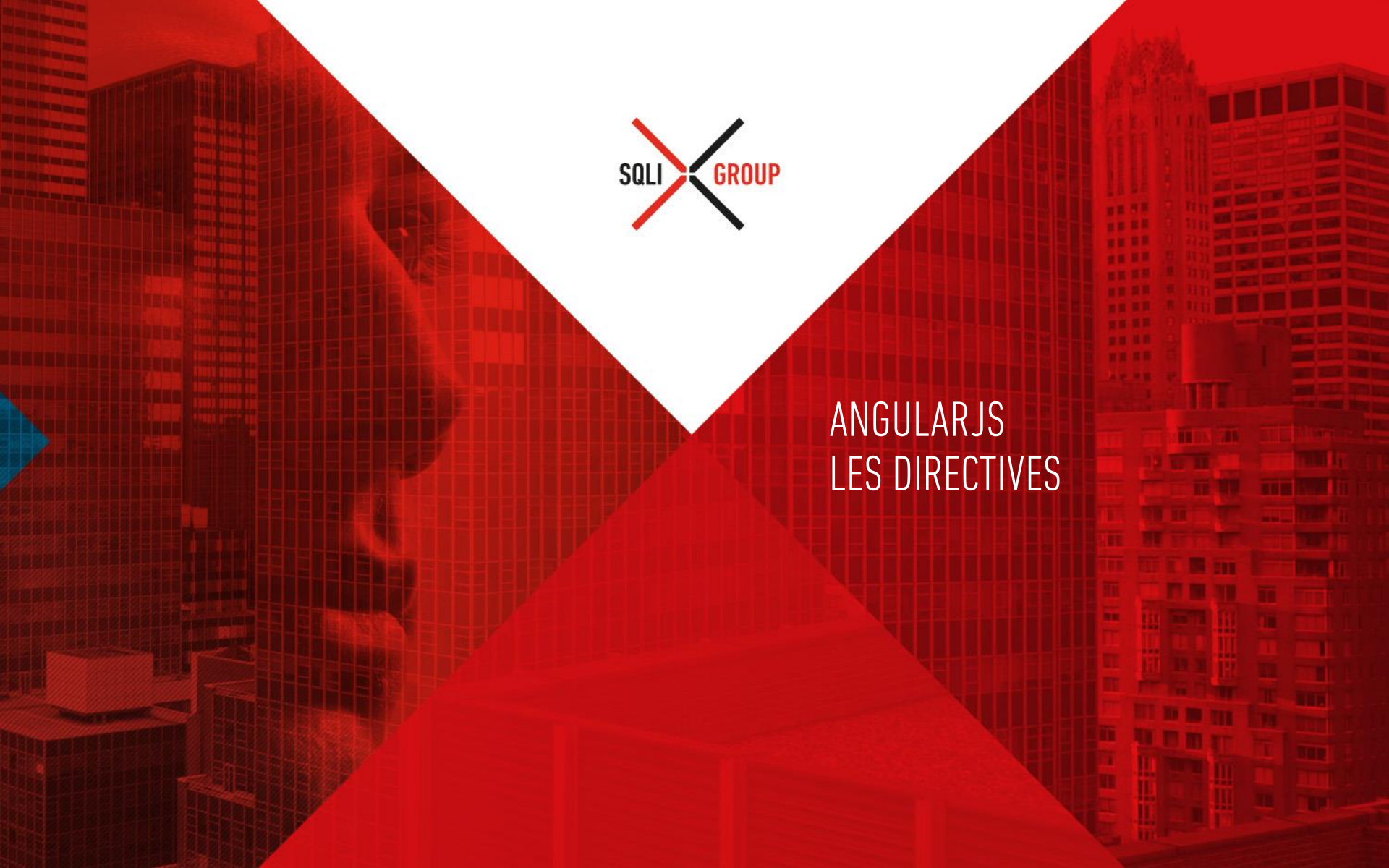


ANGULARJS
LES DIRECTIVES

SQLI GROUP

LES DIRECTIVES

DIRECTIVES

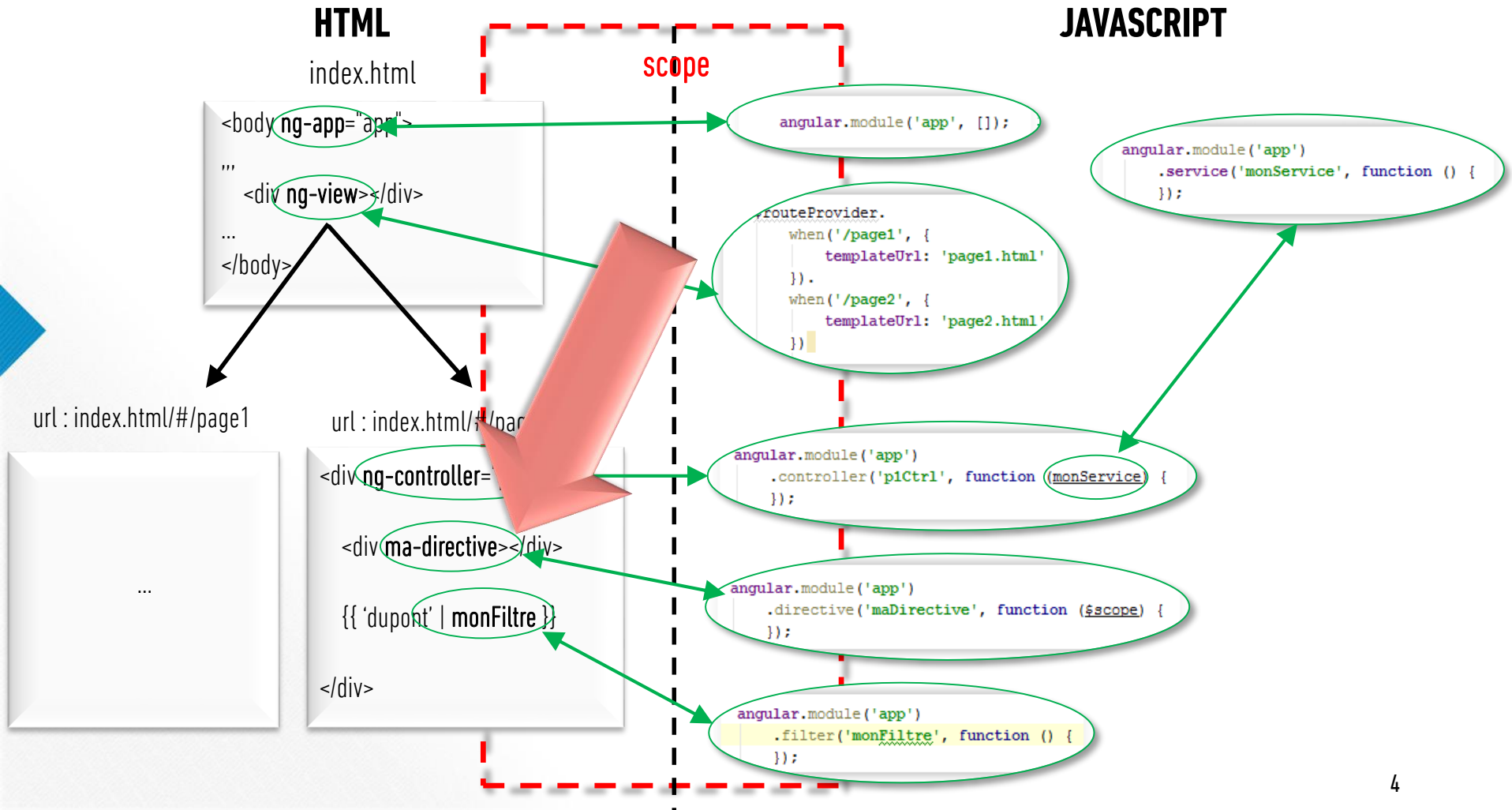
QU'EST-CE QU'UNE DIRECTIVE ?

Une directive est un composant AngularJS permettant de créer son propre vocabulaire HTML.

Possibilité d'enrichir HTML avec :

+ De nouveaux éléments : `<madirective></madirective>`
+ De nouveaux attributs : `<span madirective="exp">`
+ De nouvelles classes : `<span class="madirective:exp">`
+ Des commentaires : `<!-- directive: madirective exp -->`

ON EST OÙ?


HTML
index.html
<body ng-app="app">
...
<div ng-view>
...
</body>
scope
JAVASCRIPT
angular.module('app', []);
angular.module('app')
.service('monService', function () {
});
routeProvider.
when('/page1', {
templateUrl: 'page1.html'
}).
when('/page2', {
templateUrl: 'page2.html'
})
url : index.html/#/page1
...
url : index.html/#/pa
<div ng-controller=
<div ma-directive>
{{ 'dupont' | monFiltre }}

angular.module('app')
.controller('p1Ctrl', function (monService) {
});
angular.module('app')
.directive('maDirective', function ($scope) {
});
angular.module('app')
.filter('monFiltre', function () {
});

DIRECTIVES

QUELQUES DIRECTIVES ANGULARJS

Application
- ng-app
- ng-controller

Formulaire
- ng-pattern
- ng-minlength
- ng-maxlength
- ng-required
- ng-list
- ng-true-value
- ng-false-value
- ng-option
- ng-submit

Template
- ng-csp
- ng-disabled
- ng-hide
- ng-show
- ng-if
- ng-mouse
- ng-repeat
- ng-switch
- ng-transclude
- ng-view

Opération
- ng-change
- ng-checked
- ng-click
- ng-href
- ng-selected

Binding
- ng-bind
- ng-model
- ng-init
- ng-src
- ng-style

DIRECTIVES

NOMMAGE ET NAMESPACES

Différentes variantes de syntaxes sont possibles:

Préfixe	Format	Exemple
Aucun	namespace-name	ng-repeat="item in items"
XML	namespace:name	ng:repeat="item in items"
HTML5	data-namespace-name	data-ng-repeat="item in items"
X-tags	x-namespace-name	x-ng-repeat="item in items"

ngRepeat

Processus de normalisation :

- Suppression de `data-` et `x-`
- Conversion des mots délimités par `:` , `-` ou `_` en camelCase

DIRECTIVES

APERÇU DE L'API DE CRÉATION D'UNE DIRECTIVE

```
var module = angular.module(…);

module.directive('namespaceNomDirective', function(parametresInjectables) {
  return {
    restrict: string,
    priority: number,
    template: string,
    templateUrl: string,
    replace: boolean,
    transclude: boolean,
    scope: boolean ou objet,
    controller: function
    controllerAs : string,
    require: string,
    link: objet ou function,
    compile: function (a la place de link)
  }
}
```

DIRECTIVES

APERÇU DE L'API DE CRÉATION D'UNE DIRECTIVE

Propriétés	Description
restrict	Déclare comment la directive va être utilisée dans le template : élément, attribut, etc.
priority	Définit l'ordre d'exécution (par rapport aux autres directives placées sur le même élément).
template	Définit un template via une chaine de caractères.
templateUrl	Définit un template à charger via une URL.
replace	Si 'true' remplace l'élément courant, sinon est ajoutée à l'élément courant.
transclude	Si 'true', les éléments enfants de la directive pourront être placés dans le template de la directive
scope	Contrôle le scope utilisée par la directive (scope parent, nouveau scope, scope isolé à définir)
controller	Crée un contrôleur qui publie une API pour communiquer entre directives
require	Requiert la présence d'une autre directive (utilisation d'une API exposée par une autre directive).
link	Définit un comportement à appliquer lors de liaison de la directive au template.
compile	Définit un comportement à appliquer lors de la compilation (avant le link)

DIRECTIVES

CRÉER UNE DIRECTIVE – DONNER UN NOM A LA DIRECTIVE

+ Une directive est construite via la fonction « directive » d'un module
+ Le nom d'une directive est au format « camelCase »
+ Il est conseillé de préfixer le nom d'une directive avec un namespace pour limiter les risques de collisions avec des directives externes.

```
monModule.directive('sqliMaDirective', …)
```

+ AngularJS va permettre d'utiliser le nom défini suivant les différentes variantes des validateurs HTML. Exemple pour « sqliMaDirective » : sqli-ma-directive, sqli:ma-directive, data-sqli-ma-directive, x-sqli-ma-directive

DIRECTIVES

CRÉER UNE DIRECTIVE – PROPRIÉTÉ RESTRICT

- La propriété « restrict » spécifie les options de déclaration de la directive. Sa valeur par défaut est « A ».
- Sa valeur est composée d'un ou plusieurs caractères suivants :

Caractère	Description	Exemple
E	Élément	<ma-directive></ma-directive>
A	Attribut	
C	Classe	
M	Commentaire	<!– directive: ma-directive exp -->

- Par exemple, « `restrict : 'EA'` » indique que la directive peut être utilisée à la fois comme un élément ou un attribut.

DIRECTIVES

CRÉER UNE DIRECTIVE – PROPRIÉTÉ TEMPLATE & TEMPLATEURL

- Les « templates » dans une directive servent à créer des composants graphiques réutilisables.
- Un composant graphique est constitué d'un ensemble d'éléments du DOM, dans un seul élément racine.
- Un « template » peut être défini avec la propriété « template » via une chaine de caractères ou la propriété « templateUrl » via une URL d'accès à un « template »
- Exemples :

```
template : '<div>Mon Widget</div>'
```

```
templateUrl : 'mon-widget.html'
```

DIRECTIVES

CRÉER UNE DIRECTIVE – PROPRIÉTÉ REPLACE

+ Si la propriété « replace » vaut « true », la directive remplace l'élément du DOM sur lequel il est appliqué.

```
<ma-directive>

</ma-directive>
```

true

```
<div>Mon Widget</div>
```

false

```
<ma-directive>
  <div>Mon Widget</div>
</ma-directive>
```

EXERCICES PRATIQUES

TP11 – PREMIÈRE DIRECTIVE SIMPLE

METTRE LE MENU EN DIRECTIVE

- ✓ directive
- ✓ templateUrl

DIRECTIVES

CRÉER UNE DIRECTIVE – PROPRIÉTÉ SCOPE

La propriété scope définit la stratégie d'intégration d'une directive.

+ Si sa valeur est false, le scope parent est utilisé. C'est la valeur par défaut.

+ Si sa valeur est true un nouveau scope est créé (scope enfant). Si plusieurs directives sont appliquées à un seul élément, un seul nouveau scope sera créé.

+ Si sa valeur est un objet alors il s'agit d'un scope dit « isolé », définissant une stratégie de binding.

  ➡ C'est le moyen de créer un composant réutilisable

DIRECTIVES

CRÉER UNE DIRECTIVE – PROPRIÉTÉ SCOPE (STRATÉGIE DE BINDING)

```
monModule.directive('maDirective',
 function() {
  return {
        scope: {
            nom_attribut : 'strategie_binding',
            nom_alias_attribut : 'strategie_binding nom_attribut'
        }
}
```

Stratégie Binding	La valeur de l'attribut doit être
@	Une chaine de caractères.
=	Une propriété du scope parent (binding bi-directionnel).
&	Une fonction du scope parent qui sera appelée plus tard.

DIRECTIVES

CRÉER UNE DIRECTIVE – PROPRIÉTÉ SCOPE (EXEMPLES)

```
monModule.directive('maDirective',
 function() {
  return {
       …
       scope: true // ou false
  }
});
```

```
<ma-directive></ma-directive>
```

```
monModule.directive('maDirective',
 function() {
  return {
       …
       scope: {
           monAttrStr:  '@',
           monAttrRef:  '=',
           monAttrFunc: '&',
           monAttrAlias: '=monAttr'
       }
  }
});
```

```
<ma-directive
  mon-attr-str="Ceci est un message"
  mon-attr-ref="trip1"
  mon-attr-func="findAll(param)"
  mon-attr="trip2"
></ma-directive>
```

DIRECTIVES

CRÉER UNE DIRECTIVE – PROPRIÉTÉ TRANSCLUDE

+ La transclusion permet de réutiliser dans le « template » d'une directive des éléments enfants définis lors de son utilisation.

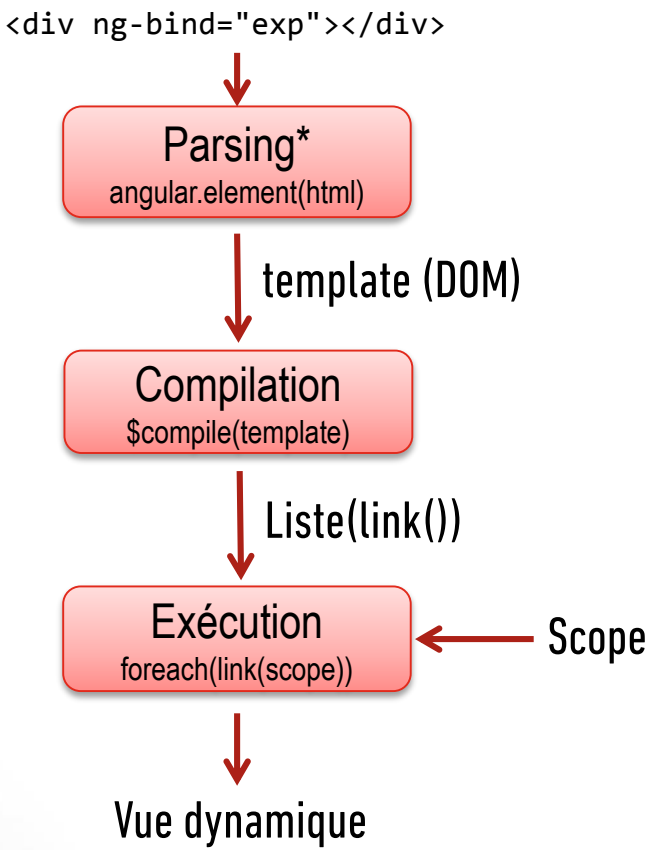
```
monModule.directive('maDirective',
 function() {
  return {
    restrict: 'E',
    replace: true,
    template:'<div>Hi,<div ng-transclude></div></div>',
    transclude: true
  }
});
```

```
<ma-directive>
  <div>Bonjour</div>
</ma-directive>
```

```
<div>"Hi,"
  <div ng-transclude>
    <div class="ng-scope">Bonjour</div>
  </div>
</div>
```

DIRECTIVES

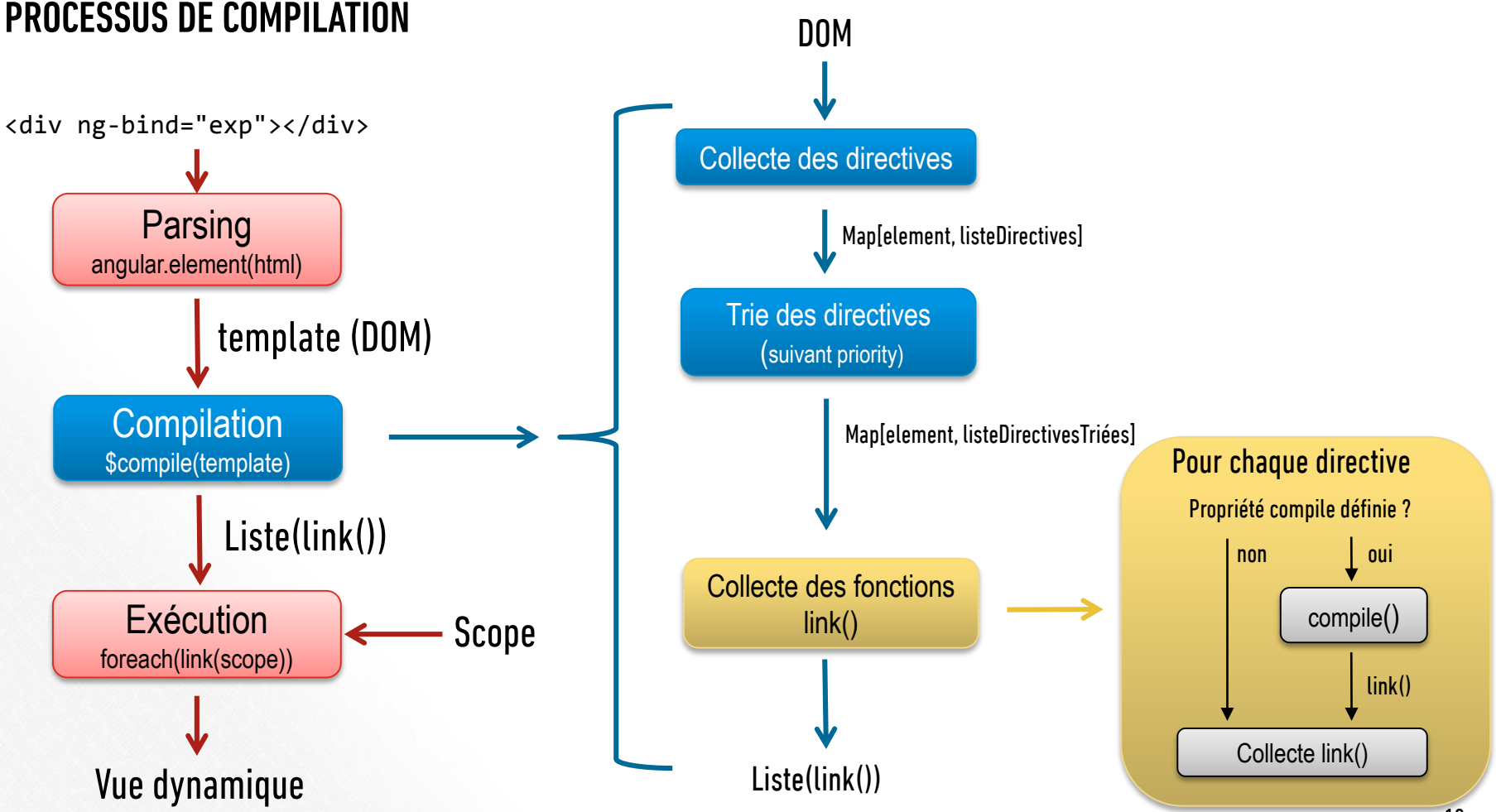
PROCESSUS DE CRÉATION D'UNE VUE DYNAMIQUE



* effectuée par le navigateur

DIRECTIVES

PROCESSUS DE COMPILATION



DIRECTIVES

FONCTION LINK

- Il y a deux catégories de fonction Link() : PreLink() et PostLink().
- PreLink() est exécuté avant que les fonctions Link() des éléments enfants ne soient exécutés. Il est risqué de manipuler le DOM dans cette fonction.
- PostLink() est exécuté après que les fonctions Link() des éléments enfants ne soient exécutés.

DIRECTIVES

PARAMÈTRES D'UNE FONCTION LINK

```
function link (scope, element, attrs, controller, transcludeFn) {
            …
}
```

- scope : le scope utilisé par la directive.
- element : l'instance de l'élément où est appliqué la directive
- attrs : liste d'attributs de l'élément où est appliqué la directive
- controller : l'instance de contrôleur si au moins une directive pour lequel la propriété controller a été définie est appliqué sur le même élément que la directive en cours de définition
- transcludeFn : une fonction de transclusion

DIRECTIVES

CRÉER UNE DIRECTIVE – PROPRIÉTÉ PRIORITY

- La propriété priority est un nombre qui définit la priorité de compilation d'une directive au sein d'un élément dans le cas où plusieurs directives sont appliquées à l'élément. Plus sa valeur est grande, plus la directive est prioritaire.
- Sa valeur par défaut est 0.
- A l'étape d'exécutions des fonctions Link(), les fonctions PreLink() sont exécutées suivant les priorités définies et les fonctions PostLink() à l'inverse de l'ordre spécifié.

```
Directive1 => priorité 500
Directive2 => priorité 700
Directive3 => priorité 400
```

```
PreLink2()
PreLink1()
PreLink3()
PostLink3()
PostLink1()
PostLink2()
```

DIRECTIVES

CRÉER UNE DIRECTIVE – PROPRIÉTÉ LINK

+ La propriété link permet de spécifier la fonction Link() à utiliser pour la directive.
+ Elle peut être définie avec un objet qui contient des fonctions PreLink() et PostLink().

```
link : {
            pre : function preLink(scope,element,attrs,controller, transcludeFn){…},
            post: function postLink(scope,element,attrs,controller, transcludeFn){…}
        }
```

+ Elle peut être définie avec une fonction. Il s'agit alors d'une fonction PostLink().

```
link : function postLink(scope,element,attrs,controller, transcludeFn){…}
```

+ Elle n'est utilisée que si la propriété compile n'est pas définie.

DIRECTIVES

CRÉER UNE DIRECTIVE – PROPRIÉTÉ COMPILE 1/2

+ La propriété compile définit une fonction de transformation du DOM du template.

```
compile: function(element, attrs) {
            …
}
```

+ element : l'élément sur lequel est appliqué la directive.
+ attrs : liste d'attributs de l'élément.

Il existait un 3e paramètre (transclude) désormais déprécié.

DIRECTIVES

CRÉER UNE DIRECTIVE – PROPRIÉTÉ COMPILE 2/2

+ Elle doit retourner une fonction de Link

Retourne un objet qui contient des fonctions PreLink() et PostLink() :

```
compile: function(element, attrs, transclude) {
          // …traitement…
         return {
          pre : function preLink(scope,element,attrs,controller, transFn){…}
          post: function postLink(scope,element,attrs,controller, transFn){…}
         }
}
```

Retourne une fonction PostLink() :

```
compile: function(element, attrs, transclude) {
          // …traitement…
         return function postLink(scope,element,attrs,controller,transFn){…}
}
```

EXERCICES PRATIQUES

TP12 – SCOPE PARENT, PUIS ENFANT, PUIS ISOLÉ

CRÉER UNE DIRECTIVE POUR L'AFFICHAGE DES CATÉGORIES DANS « HOME »

- ✓ directive
- ✓ scope

DIRECTIVES

CRÉER UNE DIRECTIVE – PROPRIÉTÉ CONTROLLER

- La propriété controller définit une fonction qui sera exécutée avant la phase de collecte des fonctions Link().
- Le contrôleur d'une directive peut être injecté dans une autre directive. Ce canal de communication est à privilégier pour faire communiquer deux directives entre elles.

```
controller: ['$scope','$element','$attrs','$transclude',…,
function($scope,$element,$attrs,$transclude, …) {
            …
}]
```

- $scope : le scope courant associé à l'élément.
- $element : l'élément courant
- $attrs : les attributs de l'élément
- $transclude : fonction de transclusion.
- … : autres éléments injectables (services…)

DIRECTIVES

CRÉER UNE DIRECTIVE – PROPRIÉTÉ REQUIRE

+ La propriété require spécifie la ou les directives prérequis au fonctionnement de la directive en cours de création. Elle est nécessaire pour injecter un contrôleur dans une fonction Link().
+ Elle peut être définie avec une chaine de caractères

```
require: 'maDirective2'
```

+ Ou un tableau de chaine de caractères

```
require: ['^directiveParent', '?directiveOptionnelle', '^?directiveParentOptionnelle']
```

+ Les directives spécifiées doivent avoir la propriété controller définie et être appliquée au même élément.
+ Le nom d'une directive peut être préfixé par ? (optionnel), ^ (recherche le contrôleur dans les scopes parents) ou ?^ (optionnel et recherche le contrôleur dans les scopes parents) .

DIRECTIVES

CRÉER UNE DIRECTIVE – PROPRIÉTÉ CONTROLLERAS

- La propriété controllerAs définit un alias du contrôleur dans le scope de la directive.
- Elle permet d'utiliser le contrôleur dans un template. Le contrôleur est alors référencé dans le template via son alias.

```
controllerAs: 'maDirectiveCtrl'
```

EXERCICES PRATIQUES

TP13 – COMMUNICATION ENTRE DIRECTIVES

DANS DASHBOARD, CRÉER UN COMPOSANT ACCORDÉON→ UNE SEULE RUBRIQUE (AUTEURS, ÉDITEURS, CATÉGORIES) D'OUVERTE

- ✓ 2 directives : accordeon & accordeon-item
- ✓ 1 contrôleur sur accordeon
- ✓ require sur accordeon-item
- ✓ utiliser le contrôleur d'accordeon depuis accordeon-item

SQLI GROUP

SCOPE

NOTIONS AVANCÉES

DIRECTIVES

SCOPE.$WATCH

Collecte une fonction à exécuter si une expression change.

```
scope.$watch(exp,listener,objectEquality)
```

- exp : une expression ou une fonction à surveiller
- Listener : une expression ou une fonction à évaluer si exp change
- objectEquality : un booléen. Si true, la détection du changement sera faite avec angular.equals.

```
scope.maVariable = 5;

scope.$watch('maVariable', function(newValue,oldValue){
   // Code exécuté si maVariable change et si scope.$digest() est appelé
});
```

DIRECTIVES

SCOPE.$EVAL

Évalue une expression dans le scope courant.

```
scope.$eval(exp, vars)
```

- exp : une expression à évaluer.
- vars : un objet qui contient des variables à ajouter ou surchager dans le scope courant.

```
scope.a = 5;
Scope.b = 10;

scope.$eval('a+b') // = 15

scope.$eval('a+b', {a : 0}); // = 10
```

DIRECTIVES

SCOPE.$DIGEST

Evalue les expressions de tous les écouteurs du scope courant et du scope enfant.

Déclenche des traitements définis dans les écouteurs si la valeur de l'expression correspondante à évoluer.

```
scope.$digest()
```

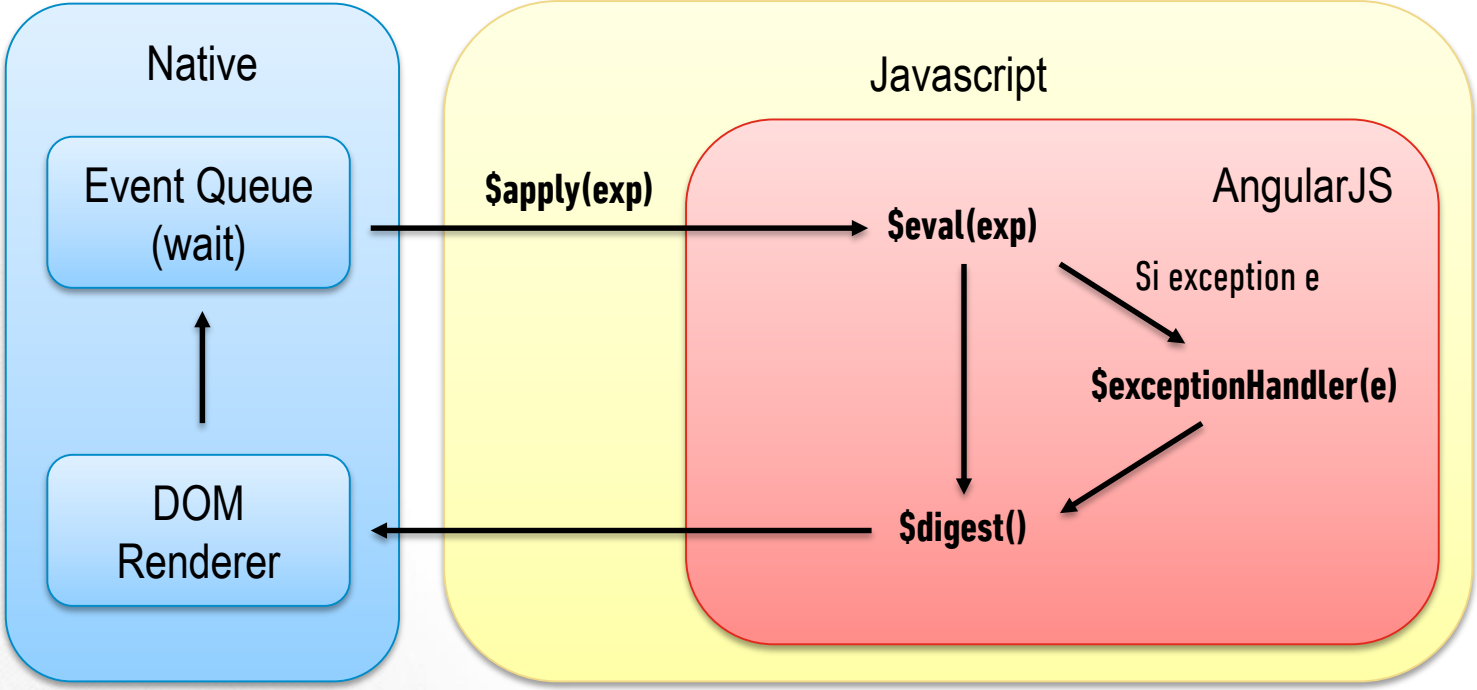
Il est recommandé de ne pas directement l'utiliser et de passer par $apply.

DIRECTIVES

SCOPE.$APPLY

$apply est utilisé pour exécuter une expression dans le contexte AngularJS depuis un contexte externe.

```
scope.$apply(exp)
```



FIN