IHM avancées

Alexandre.Demeure@inria.fr
Sybille.Caffiau@imag.fr

M2 MIAGE - 2016

Au programme cette année

Etude de technologies pour l'implémentation d'IHM

- Angular 2
- Série d'exposés sur d'autres technologies

Design et outils de maquettage

- Responsive Design
- Flat Design
- Material Design
- Cogtool
- Axure

Au programme cette année

- Aujourd'hui : Angular 2 + Choix des sujet de projet
- 29/09 : Projet
- 03/10 : Cours design et maquettage
- 06/10 : Projet
- 03/11 : Premier exposés (30 mns / groupe)
- 07/11 : Projet
- 05/12 : Projet
- 09/01 et 12/01 : Second exposés
- 12/01 : Examen individuel
  (Questions sur les cours et les exposés)

Objectifs

- Acquérir une vue d'ensemble des technologies pour le développement d'IHM (centrées web)
- Vous former à des technologies web couramment utilisées dans le développement d'applications interactives
- Apprendre à vous auto-former sur de nouvelles technologies
- Apprendre à présenter de nouvelles connaissances techniques, à les comparer

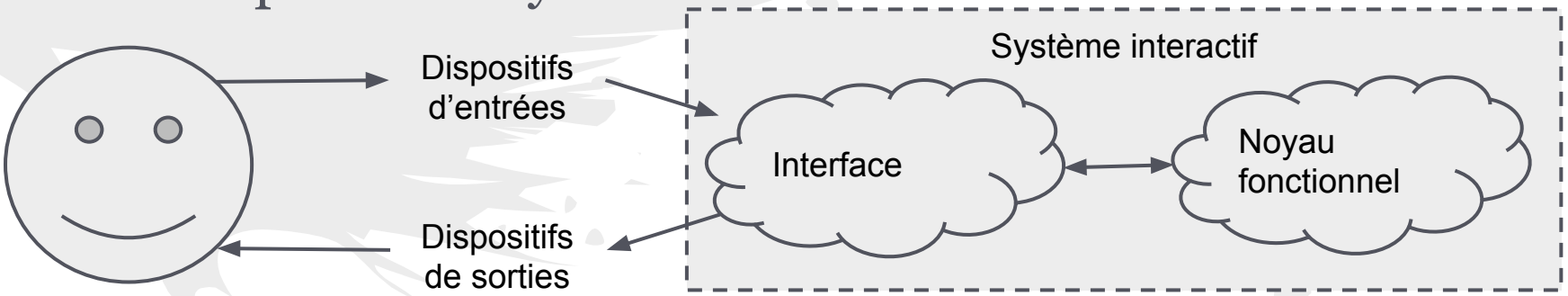
Architecture logicielle

Etude de cas
La liste de choses à faire

Architecture logicielle

Principes

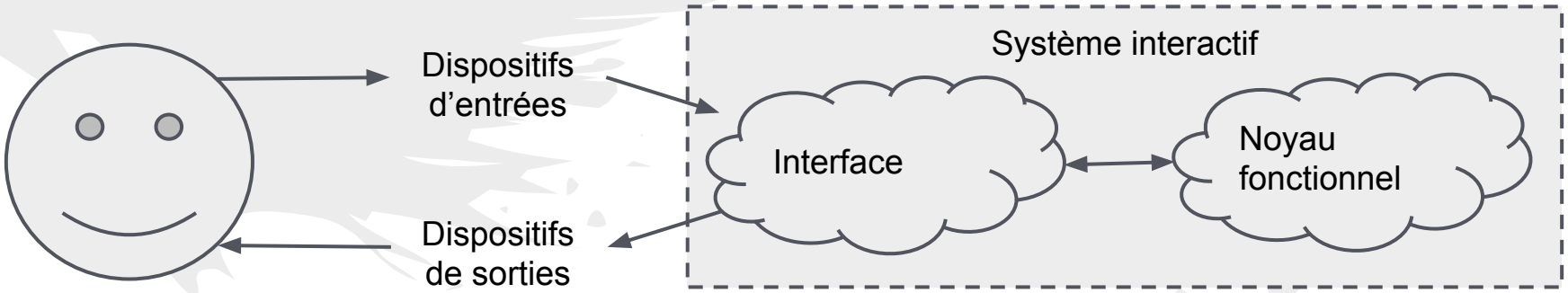
- Séparer le noyau fonctionnel de l'IHM



Architecture logicielle

Principes

- Séparer le noyau fonctionnel de l'IHM



- Noyau fonctionnel
  - ne connaît pas directement l'IHM (peut savoir qu'il y en aura une)
  - fournit des moyens pour l'IHM (annulation, erreurs, notifications, ...)
- IHM
  - connaît le noyau fonctionnel
  - utilise les moyens d'annulation, gestion d'erreurs et notifications

Architecture logicielle

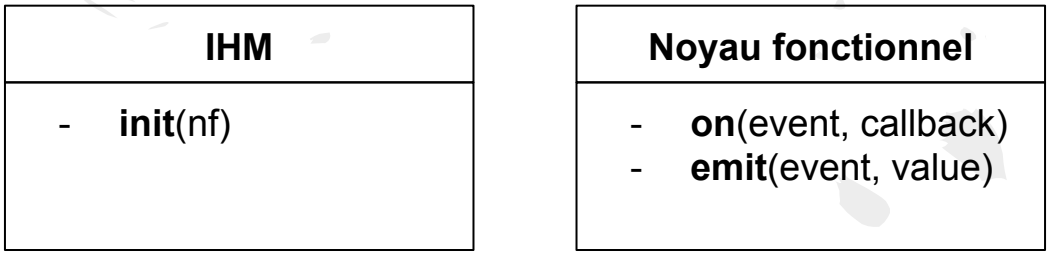
Le NF offre des services nécessaires à l'interaction

- Notification (Patron de conception "Observer")
- Prévention des erreurs (L'appel est-il licite?)
- Annulation (Etats précédents cohérents)

Architecture logicielle

Séparer les préoccupations

- Noyau fonctionnel indépendant de l'interface
- Noyau fonctionnel offre des moyens d'être observé
- Mécanisme événementiel ⇔ Publish/Subscribe



Architecture logicielle

Exemple avec une liste de choses à faire

- Noyau fonctionnel ?
- IHM ?

IHM
- **init**(nf) - ???

Noyau fonctionnel
- **on**(event, callback) - **emit**(event, value) - ???

Nouvelle tâche	
○ Tâche 1	X
○ Tâche 2	X
○ Tâche 3	X

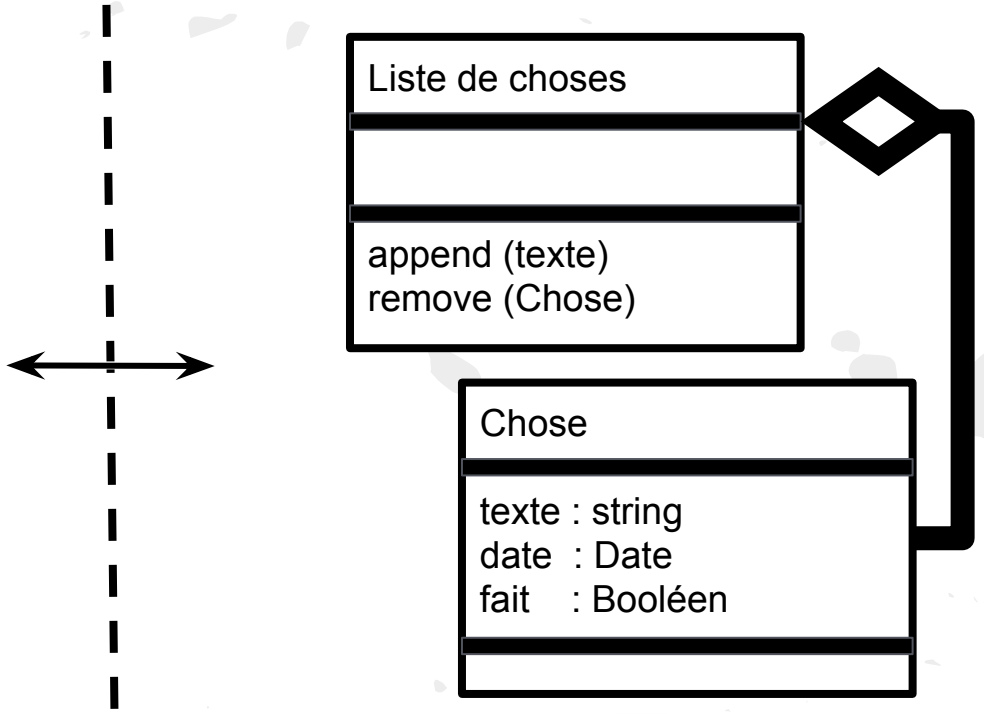
Architecture logicielle

Exemple avec une liste de choses à faire
(voir le site http://todomvc.com)

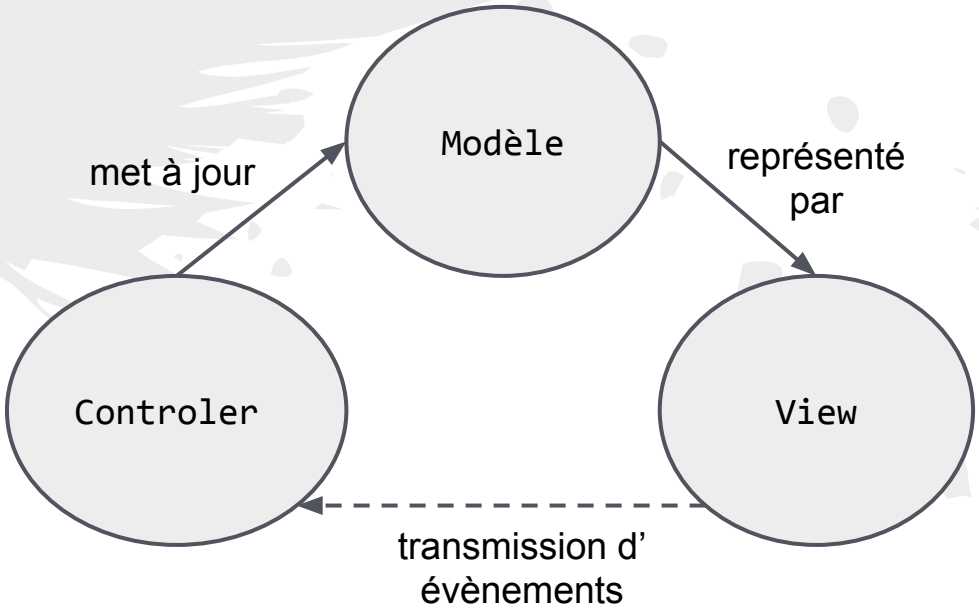
IHM

Nouvelle tâche	
○ Tâche 1	X
○ Tâche 2	X
○ Tâche 3	X

Noyau Fonctionnel (NF)



Architecture logicielle : MVC



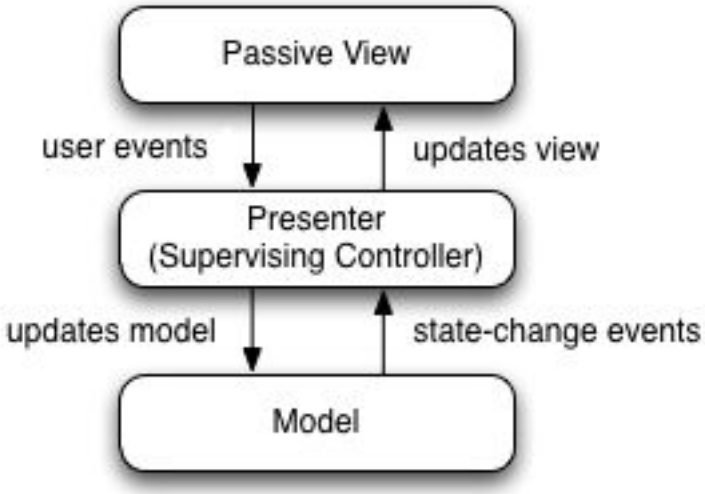
Architecture logicielle : MVC

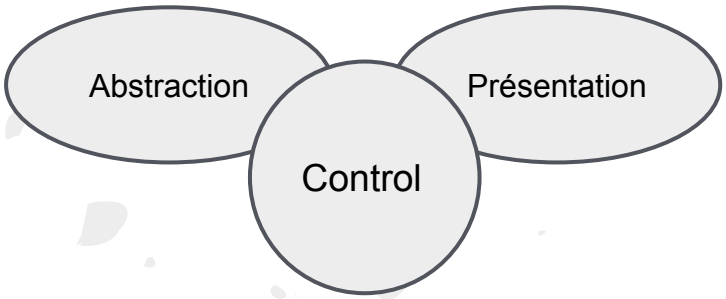
Exemple de la liste de choses à faire

Architecture logicielle : MVC

Critique de MVC pour les clients web

Architecture logicielle : MVP/PAC





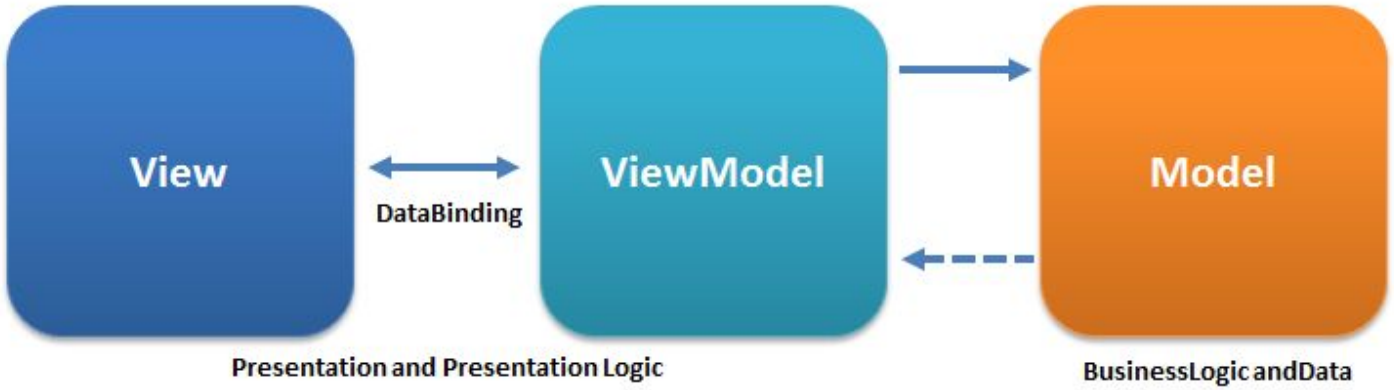
Architecture logicielle : MVP/PAC

Exemple de la liste de choses à faire

Architecture logicielle : MVP/PAC

Critique de MVP/PAC pour les clients web

Architecture logicielle : MVVM



Architecture logicielle : MVVM

Critique de MVP/PAC pour les clients web

Etude de cas
Angular 2

Etude de cas : Angular2

- Contexte
  - Qui a développé cette technologie, quand ?
  - Buts de la technologie, problèmes adressés ?
  - Qui utilise cette technologie, communauté? Projets phares ?
- Framework
  - Architecture, langages
  - Les concepts
  - Exemples illustrant la technologie
- Positionnement
  - Positionnement vis à vis des autres technologies
  - Avis personnel
  - Pourquoi est ce intéressant ?
- Démonstration

Angular 2 : Le contexte

AngularJS (v1) développé chez Google en 2009

- Projet annexe à l'origine (20% temps libre)
  - Pour les designers web (pas vraiment les devs)
  - Interagir facilement avec le back et front-end
- Projet Google Feedback (dev en GWT)
  - 6 mois de développements
  - 17.000 lignes de codes
  - Difficile à tester, à faire évoluer
- Un des développeur fait un pari
  - Google Feedback en 2 semaines avec Angular
  - Pari perdu : 3 semaines. 17.000 => 1500 lignes
  - Plus clair, testable, plus facile à faire évoluer
  - Utilisation en interne puis en externe

Angular 2 : Le contexte

AngularJS (v1)

- Philosophie : Que serait HTML si il avait été conçu pour décrire des applications ?
- Cible les Single Page Application
- Double way data binding, testabilité, injection de dépendances, ...
- Communauté très importante pour AngularJS
  - Cf google trends
  - Cf github
- Nombreux projets réalisés
  - https://www.madewithangular.com/

Angular 2 : Le contexte

Angular 2 développé PAR Google

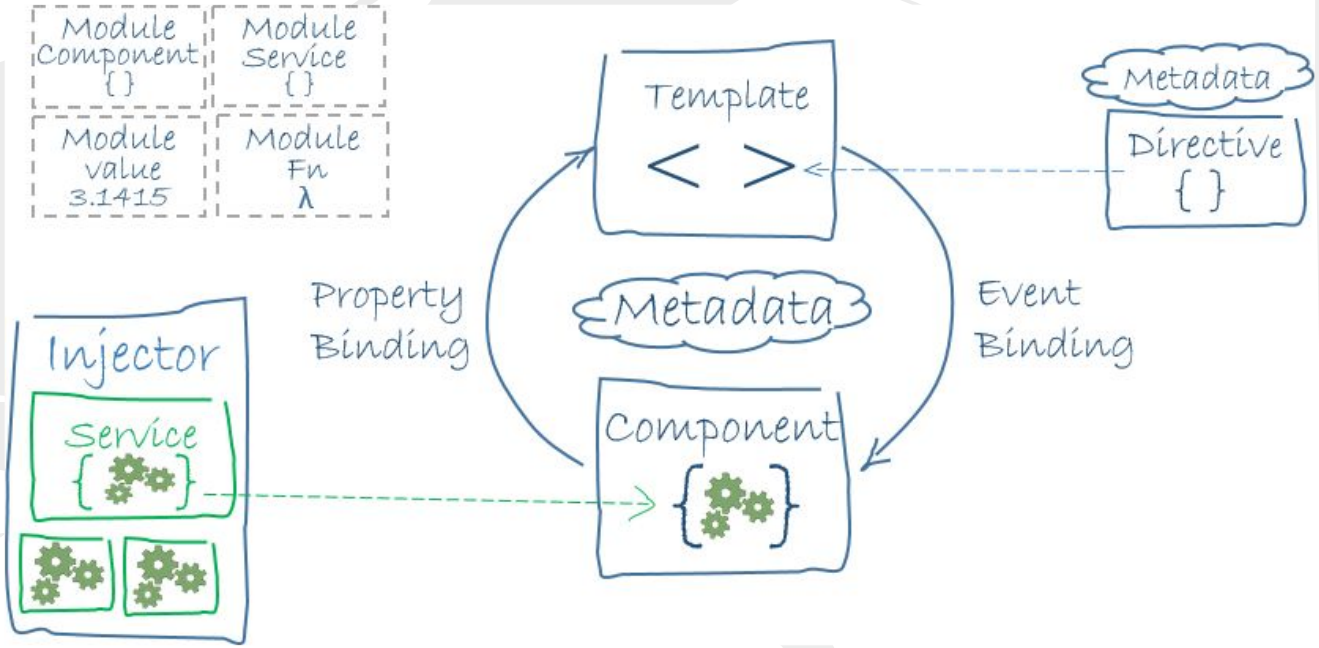
- Annoncé ~2015
- Problèmes de performances, passage à l'échelle
- EcmaScript -> TypeScript
- Standard des web components
- Angular 2.0.0 produit ~15/09/2016

Angular 2 : les concepts

- Mega framework (tout en un)
- On reste centré sur le HTML
- Augmentation du HTML avec des directives:
  - Attributs
  - Balises
  - Comportement définit par du code
- Architecture dites MV* (proche de MVVM)
- Passage à TypeScript
  - Typage fort
  - Annotations

Angular 2 : les concepts

- Modules
- Components
- Templates
- Metadata
- Data binding
- Directives
- Services
- Dependency injection



Angular 2 : Concepts

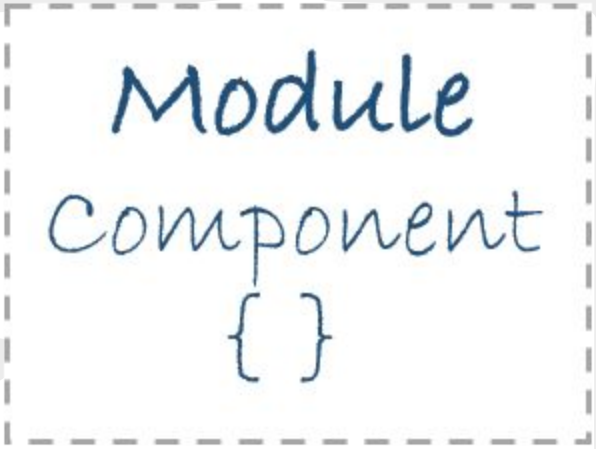
Principes, Archi, langage

Concepts centraux dans Angular

Exemples illustrant l'utilisation de cette techno de façon pertinente

Mega framework (tout en un)
Redéfinition de balises ou d'attributs qui seront interprétés

Angular 2 : Concepts



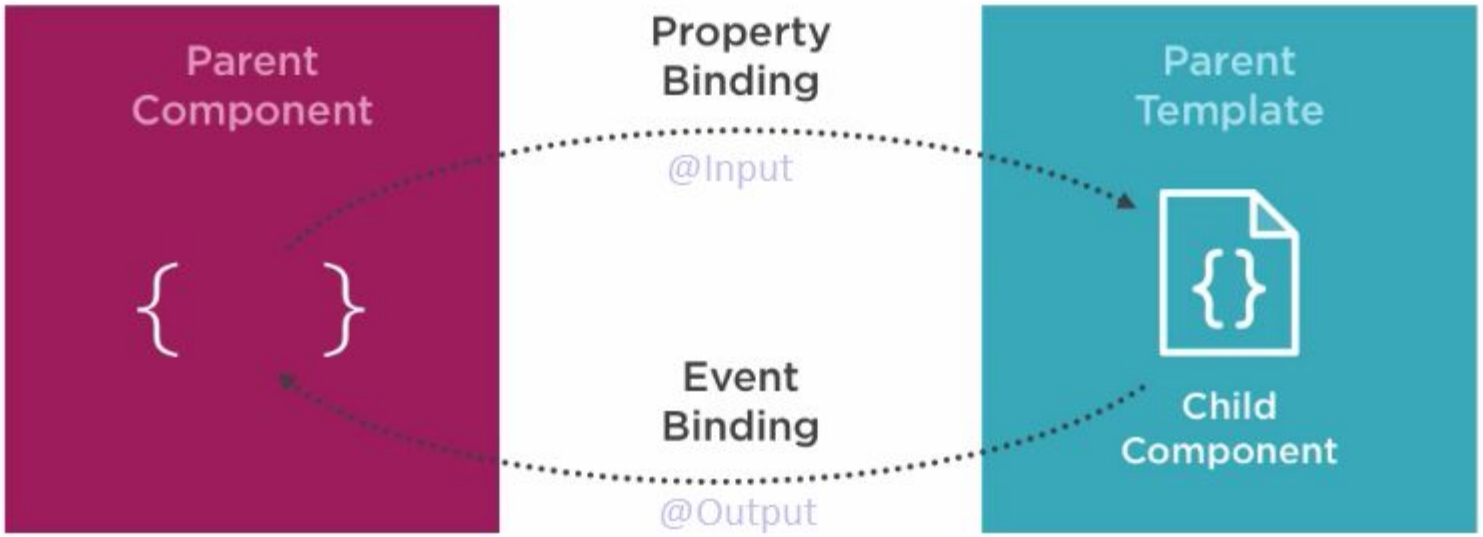
Modules

- Aggrège des composants, des directives et des services
- Différent des modules ES
- Import d'autres modules
- Export de déclarations de composants et directives
- Ensemble de services qui seront utilisés pour le module

Angular 2 : Concepts

Components

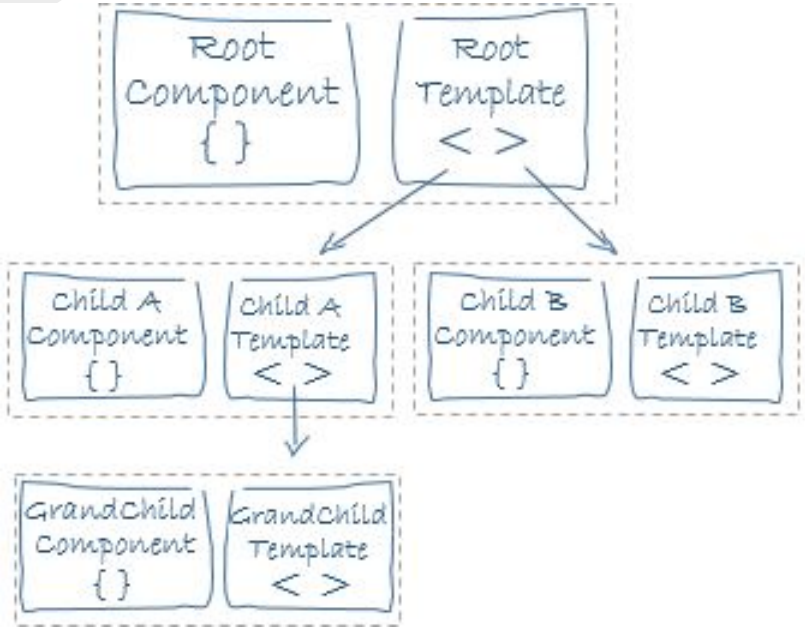
- Définition d'une nouvelle balise HTML
- Associé à une classe typescript
- Peut prendre des propriétés/attributs
- Input (property binding)
- Output (event binding)



Angular 2 : Concepts

Template

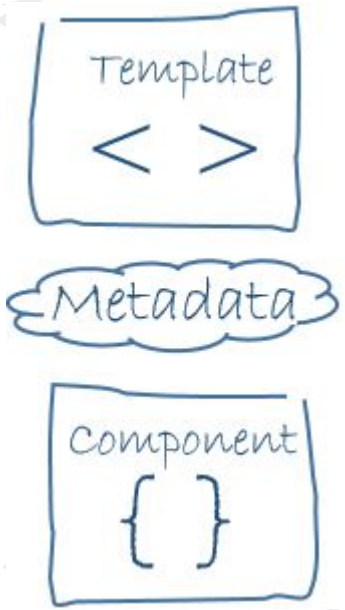
- Associé à un composant
- Indique
  - Comment représenter l'état du composant
  - Utilise les templates des composants liés



Angular 2 : Concepts

MetaData

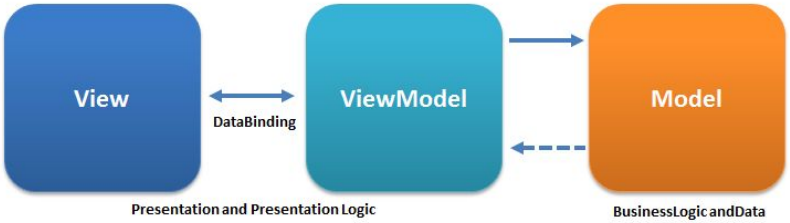
- Associe des données à une classe
- Meilleur structuration du code
- Utilisation des annotations TypeScript
- Exemple pour un composant :
  - Son template
  - Les services utilisés
  - Le nom de la balise HTML

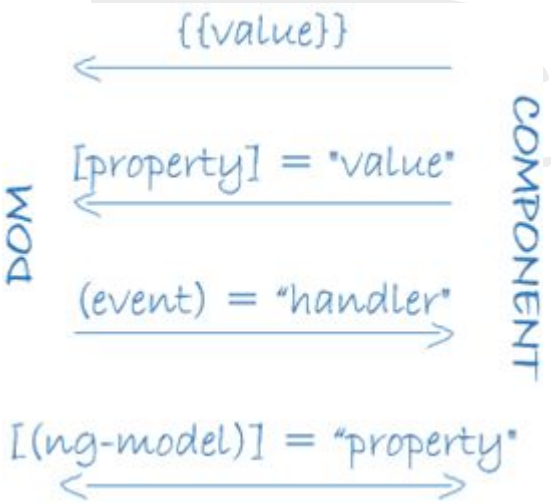


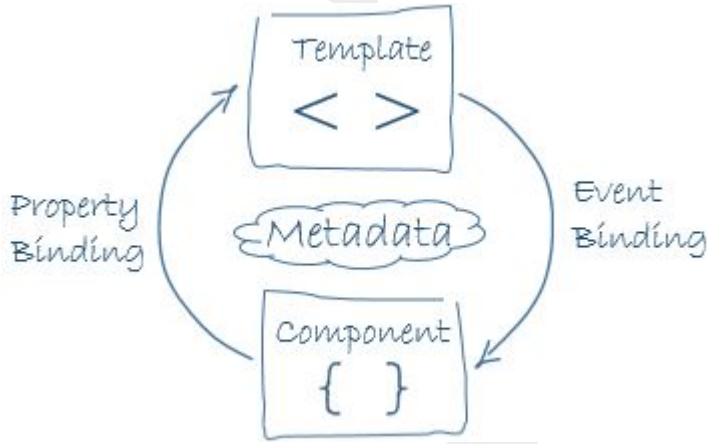
Angular 2 : Concepts

Data binding

- HTML <-> TS



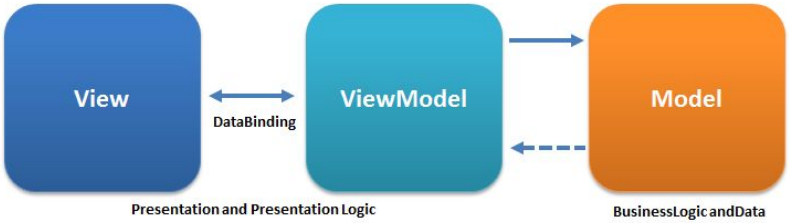


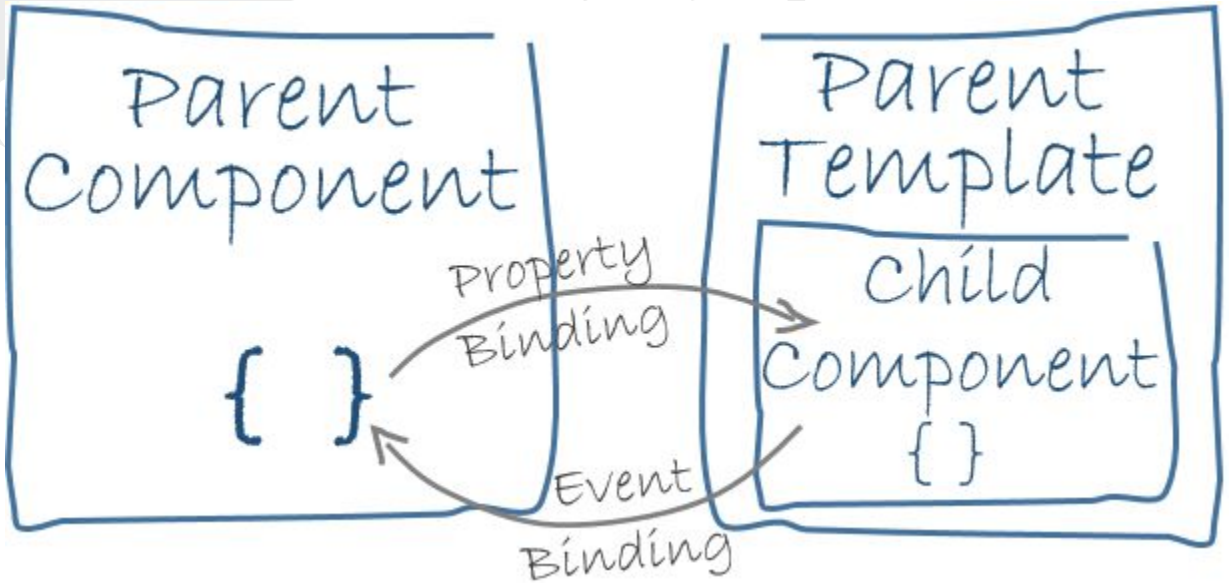


Angular 2 : Concepts

Data binding

- HTML <-> TS
- Lien entre composants





Angular 2 : Concepts

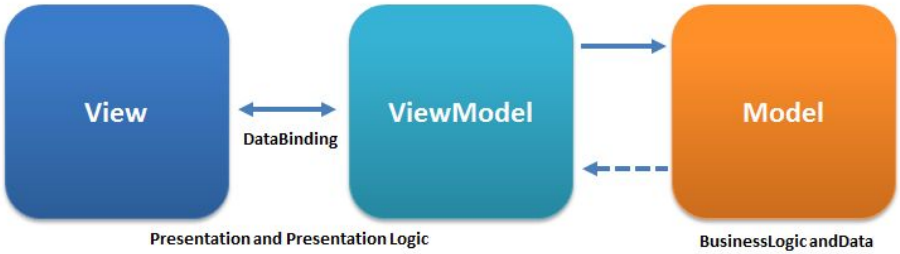
Directives

- Associée à une classe
- Indique à Angular comment interpréter les template
- Un composant est une directive avec template
- Deux autre types de directives
  - Structurelle (ex: *ngFor, *ngIf)
    <li *ngFor=”let c of choses”>...
  - Attribut
    <input ([ngModel])="nom">

Angular 2 : Concepts

Services

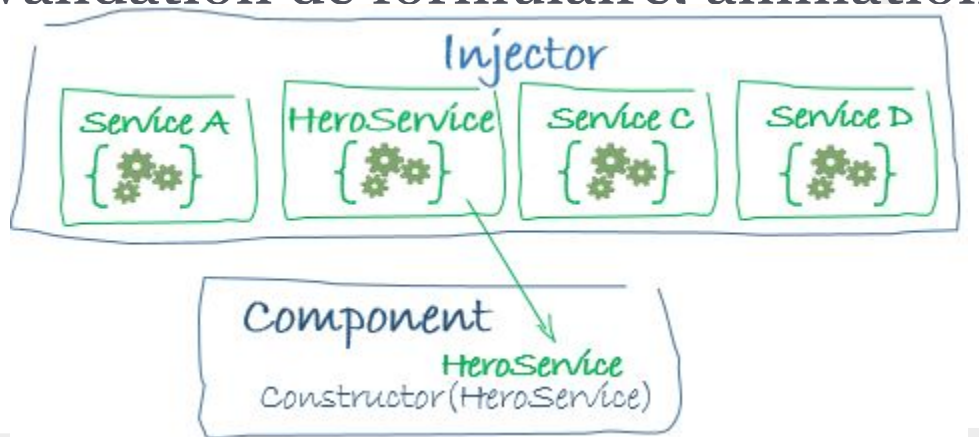
- Tout ce dont votre application peut avoir besoin
  Un nombre (ex PI), une classe, une fonction, …
  S'identifier, bus à message, …
- Permet de bien séparer l'accès au noyau du rendu
- Service : Accès au noyau fonctionnel
- Composant : utilise les services



Angular 2 : Concepts

Dependency injection

- Permet de fournir les services aux classes
- Angular détermine les services à injecter à l'aide de la signature du constructeur de la classe:
  ex: constructor(listeNF: ServiceListe) {...}
- Les services à injecter sont référencés dans le module
- Angular fourni de très nombreux services de bases : HTTP, validation de formulaire. animation, tests, ...



Liste de choses à faire

Démonstration:

- git clone https://github.com/AlexDmr/MIAGE-TodoList.git
- Bases
- Tests

Angular 2 : Positionnement

Vis à vis des autres framework :

- à vous de le dire dans vos projets

Avis personnel :

- On a tout sous la main
  (aspect mega framework)
- Typescript
- Se justifie pour des projet gros ou moyen
- Encore (très) jeune mais un grand intérêt
- Communauté !
- Support (c'est google)

Projet
Choix du sujet
Formation des groupes

Au programme cette année

- Aujourd'hui : Angular 2 + Choix des sujet de projet
- 29/09 : Projet
- 03/10 : Cours design et maquettage
- 06/10 : Projet
- 03/11 : Premier exposés (30 mns / groupe)
- 07/11 : Projet
- 05/12 : Projet
- 09/01 et 12/01 : Second exposés
- 12/01 : Examen individuel
  (Questions sur les cours et les exposés)

Sujets possibles du projet

- ReactJS
- Meteor
- Polymer / web components
- EmberJS
- MarionetteJS
- Aurelia
- KnockoutJS
- Progressive web app
- IONIC / IONIC2 / Phonegap
- NativeScript
- ...
- Toute autre technologie pour réaliser des IHM
  QT, GWT, Android, IOS