

- Pré-requis
  - Cours Java SE ZZ2
  - Objets / UML / Design Patterns
  - Cours SGBD
  - Une pincée de web …
- Évaluation
  - Je sais pas encore

Philosophie ?

- Faire des applications [WEB] professionnelles avec JAVA et une base de données
- Plus simple que Java EE / Jakarta EE
  - Partage de certaines JSR
- Convention plutôt que configuration
  - Annotation (décorations)
  - Fichiers `.properties`
  - XML ?
- Gestion du cycle de vie des objets
  - Injection de dépendances

Ecosystème ? Concurrents ?

- J2E / Java EE / Jakarta EE 1999
- Spring 2003
- Spring Boot 4.0 2014

Version 2	Version 3-4
Java 8->19 Java EEE 8	Java 17->19 Jakarta EE 9

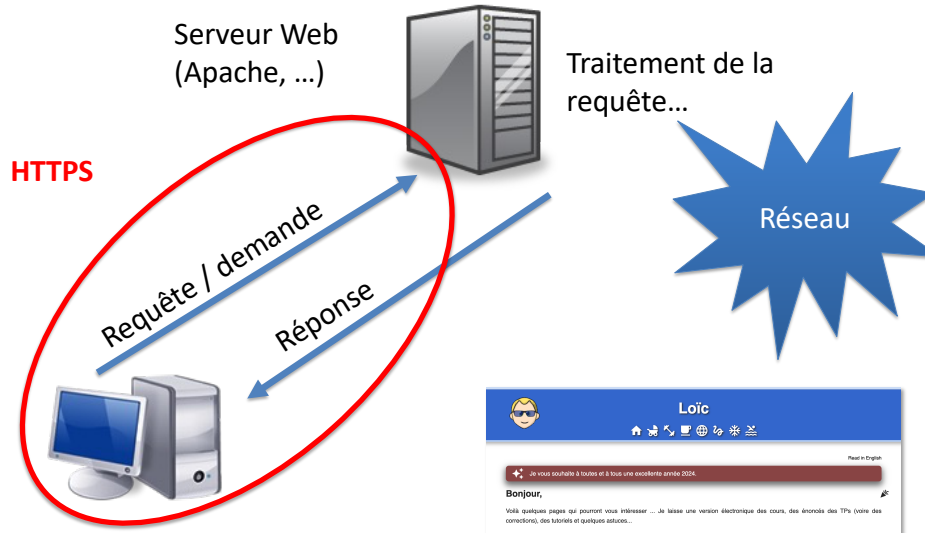
https://spring.io

UNE PINCEE DE WEB





CLERMONT AUVERGNE INP Isima



Que renvoie le serveur ?
Que comprend le navigateur ?

- Du texte
  - HTML
  - Style (CSS)
  - Javascript
- Du binaire
  - Images (format ?)
- Mais encore
  - XML
  - JSON
- Du binaire
  - Multimédia
  - Données ?

Exemple de contenu "statique"

```
<!doctype html>
<html>
 <head>
   <title> Page du loïc </title>
 </head>
 <body>
  <p>Bienvenue sur ma page</p>
 </body>
</html>
```

Envoyé par le serveur Web
Sans traitement

Traduit par le navigateur

Fichier texte : index.html

Interactions basiques

- Demande simple : GET
  - Obtention de ressource
  - Communication non sécurisée par passage par paramètre

`https://monsite.fr`

`https://monsite.fr/index.html?nom=yon&prenom=loic`

- Envoi de données : POST
  - Formulaire
  - Nécessite un traitement côté serveur

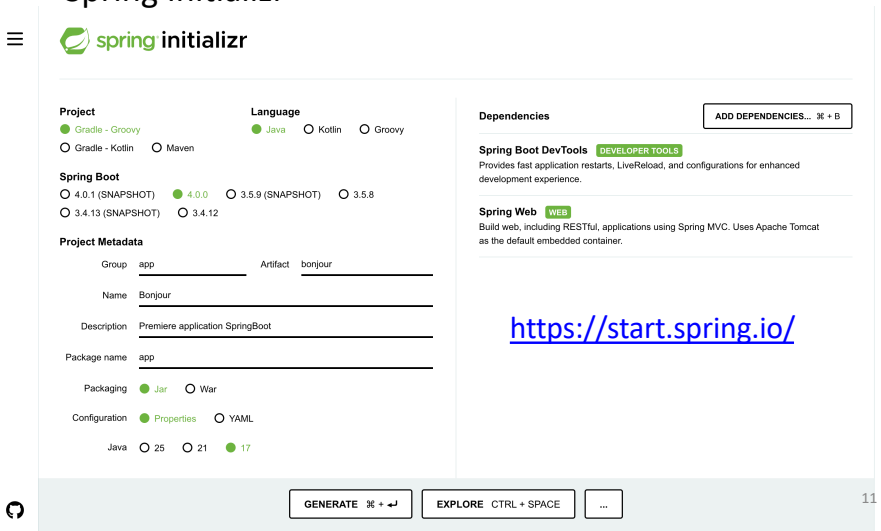
Nom
Prénom
OK

Affichage d'un page statique index.html

- Création d'un premier projet
  - Spring Initializr
- Lancement du serveur
  - Erreur à l'exécution ? configuration
- Affichage de la première page web statique
  - Page de fallback

Outils ?

- Maven / Gradle
- Spring Initializr



https://start.spring.io/

Tâches Gradle

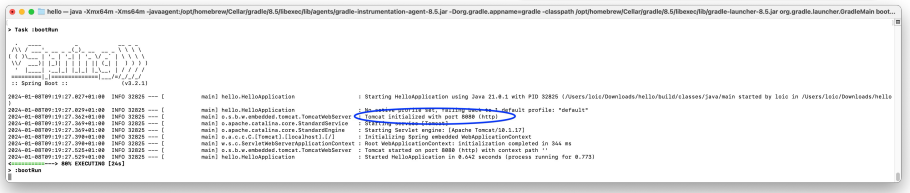
- gradle[w] build

Gestion des dépendances

Production "exécutable"

- gradle[w] bootRun

Exécution

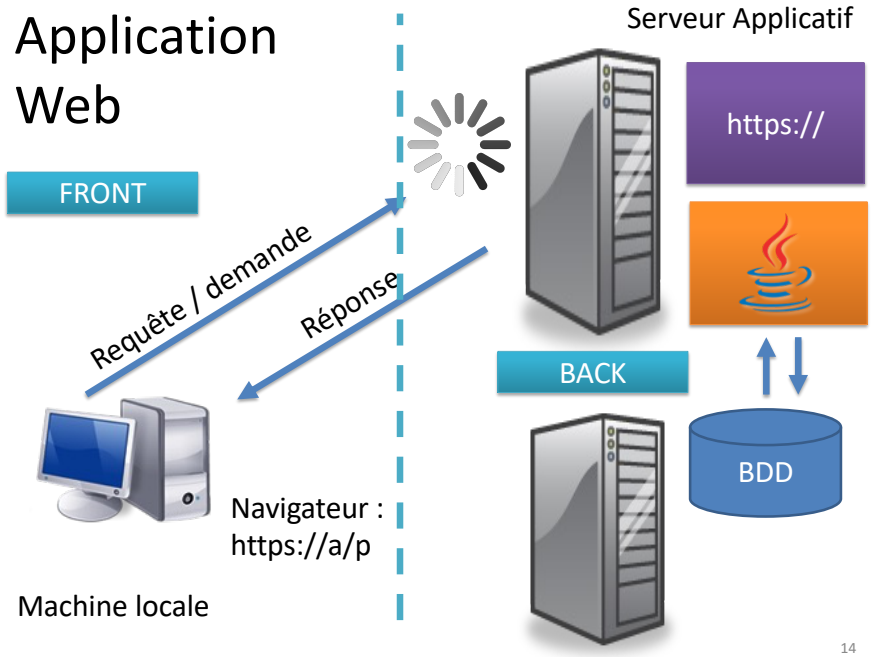


PRINCIPES DE BASE





CLERMONT AUVERGNE INP Isima



Contenu "dynamique"

- Contenu généré à la volée
  - JS, JAVA, PHP , Python, …
- Mais pas limité aux pages web
- Interaction avec d'autres éléments ?

- Demandes au serveur
- Réponses
- Servlets
- Templates

Requêtes (verbes HTTP)

- URI + Verbe => Code de retour
- Entête (head) + Corps (Body)

C	POST	• Création de ressource • Formulaire
R	GET	• Obtention d'une ressource • Page Web
U	PUT [PATCH]	• Modification de ressource [partielle]
D	DELETE	• Suppression

Quelques codes de retour

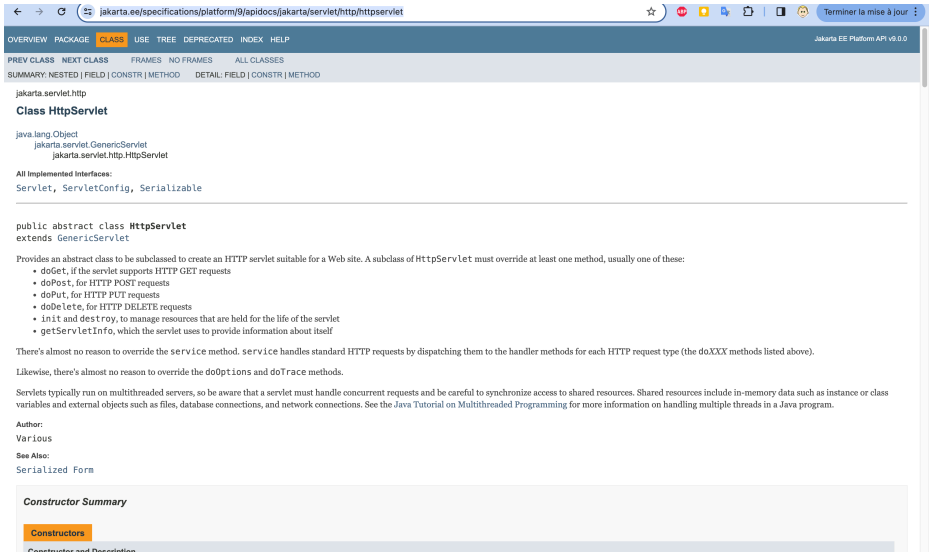
- **200 : OK**
- 201 : Created
- 204 : No content
- 400 : Bad request
- 404 : Not found
- 405 : Method not allowed
- 500 : Internal server

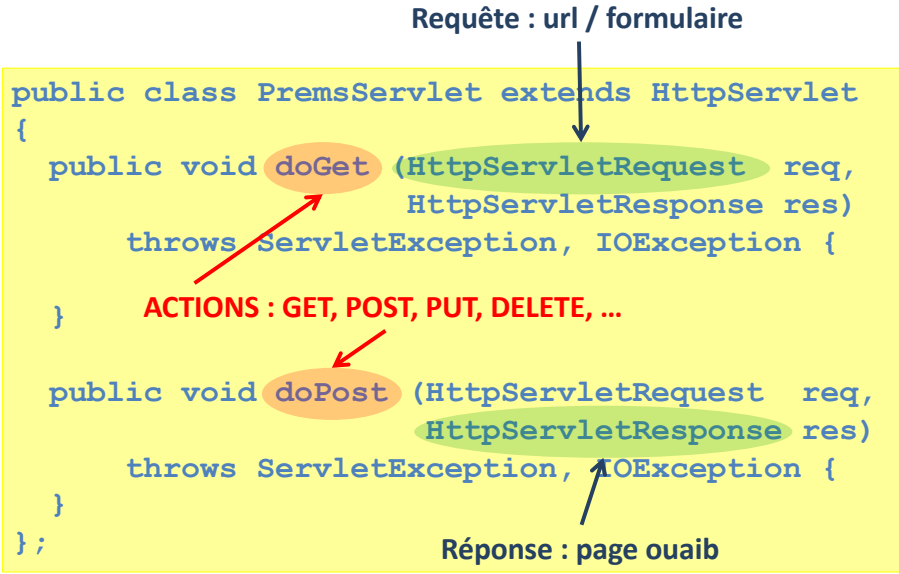
Servlet

- Classe java qui s'exécute grâce à un moteur de servlet / conteneur Web
- Historiquement, un des premiers types de composants JAVA
- Toujours utilisé pour la configuration
- Ou les traitements plus complexes

- Cycle de vie ?
  - init, service, destroy

https://jakarta.ee/specifications/platform/9/apidocs/jakarta/servlet/http/httpservlet







Première servlet

- Appel de la première servlet
- Manipulation d'un attribut
  - Tester avec différents navigateurs
- Affichage de paramètres en GET ou POST

Créer une page web ?



Designer UX/UI

Designer UX/UI
ou dév ?

NOM	PRENOM	ADRESSE	MAIL	TELEPHONE

```
<table>
  <tr>
    <td></td>
    <td></td>
  </tr>
</table>
```

NOM	PRENOM	ADRESSE	MAIL	TELEPHONE
Yon	Loïc	ISIMA	l@isima.fr	0473405042

Designer UX/UI
ou dév

Langage ?



Page dynamique HTML

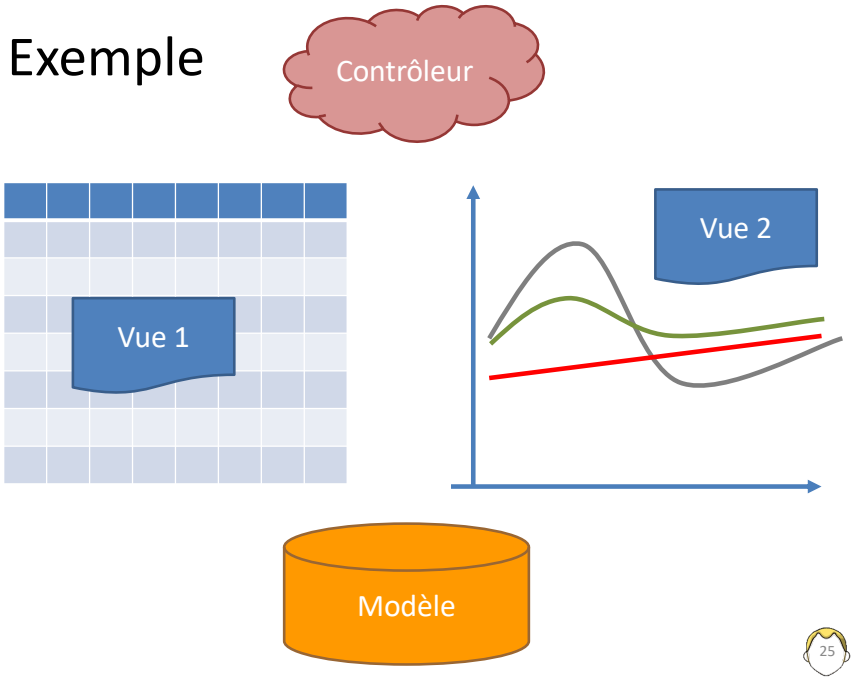
- Moteur de template ThymeLeaf
  - Enrichissement des balises HTML
  - Emplacement particulier
- Mise en place d'un contrôleur
  - Spring Web MVC
  - DispatcherServlet

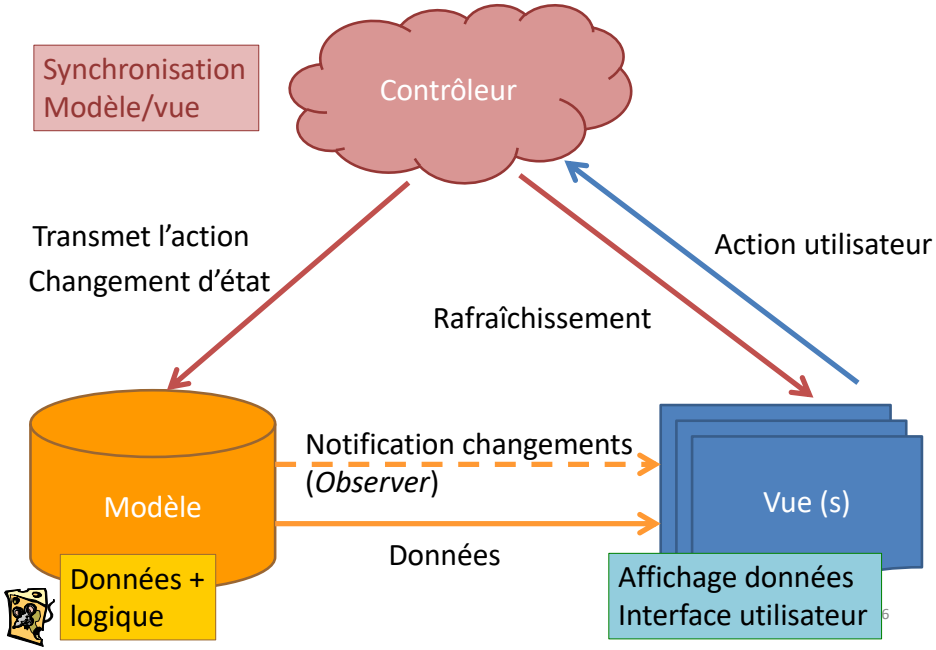
Modèle MVC

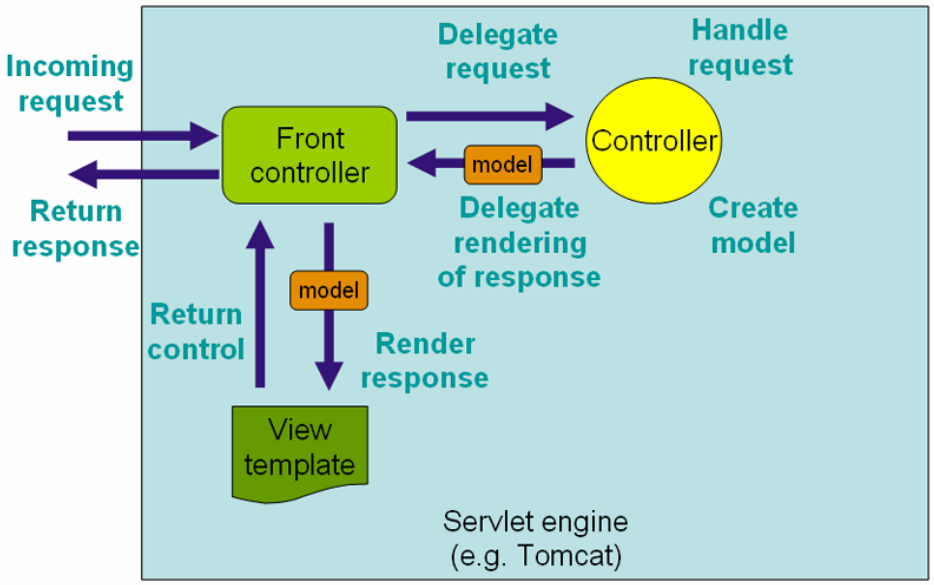
- À la sauce web
- Avec un contrôleur principal (front controller pattern)
- Modèle
  - Des POJOs
  - Gestions des requêtes par annotation
- Vue
  - Des pages web le plus souvent
- Contrôleurs
  - Comment gérer les interactions utilisateurs
  - Requêtes

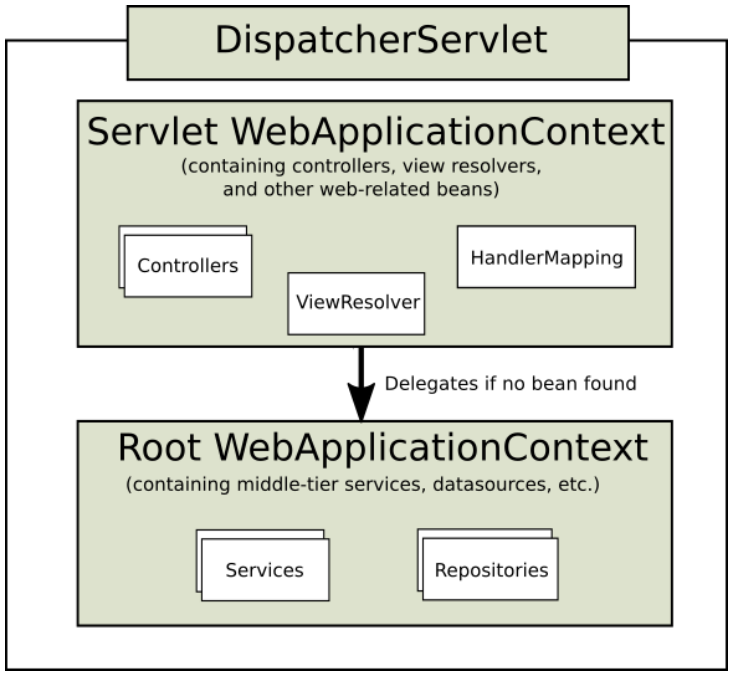
https://docs.spring.io/spring-framework/reference/web/webmvc.html

Exemple


Contrôleur
Vue 1
Vue 2
Modèle


Synchronisation
Modèle/vue
Contrôleur
Transmet l'action
Changement d'état
Action utilisateur
Rafraîchissement
Modèle
Notification changements
(Observer)
Données
Vue (s)
Données +
logique
Affichage données
Interface utilisateur

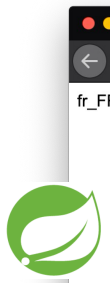

Incoming
request
Return
response
Front
controller
Delegate
request
Handle
request
Controller
model
Delegate
rendering
of response
Create
model
model
Return
control
Render
response
View
template
Servlet engine
(e.g. Tomcat)

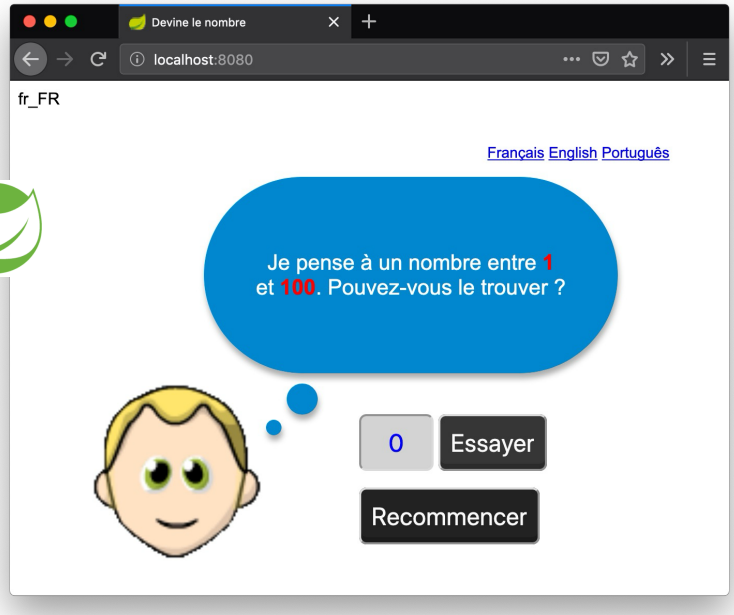

DispatcherServlet
Servlet WebApplicationContext
(containing controllers, view resolvers,
and other web-related beans)
Controllers
ViewResolver
HandlerMapping
Delegates if no bean found
Root WebApplicationContext
(containing middle-tier services, datasources, etc.)
Services
Repositories



Devine le nombre

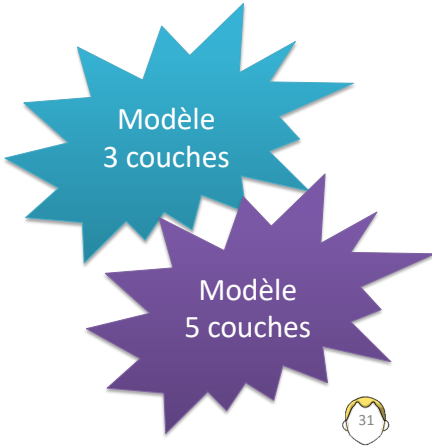
- Mise en place du formulaire et de l'interaction
  - Message « plus grand », « plus petit », « trouvé »
  - Compteur d'essais
- Gestion des erreurs sur la saisie
- Internationalisation





Une application ?

- Couche de données (persistance)
  - Entité et bases de données, ORM
- Couche métier
  - Java : Beans
- Couche présentation
  - , …
- Interopérable ?



REST ?

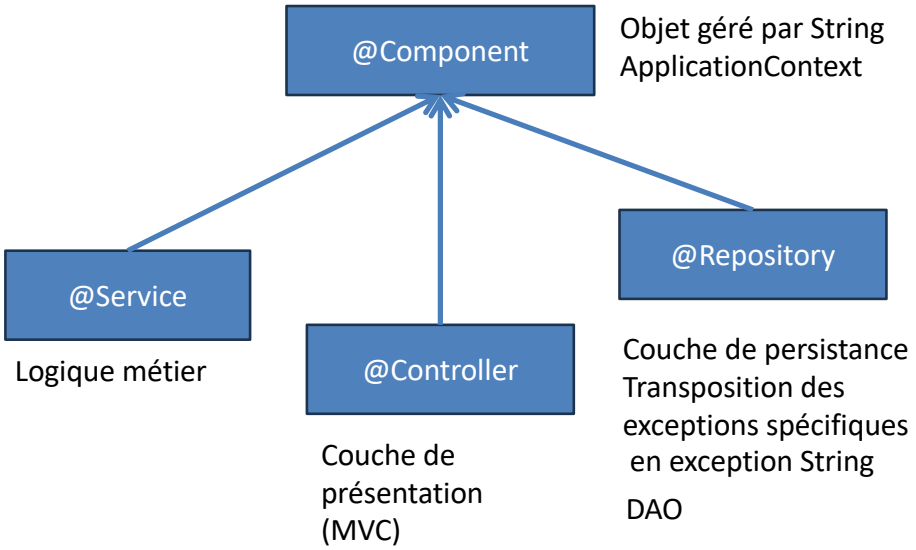
- Interaction page web / API
- Verbes + URL

- https://localhost/index.php?name=loic&qualite=rien
- https://monsite/users
- https://monsite/user/32

Modèle de Richardson

- Evaluer le niveau des API REST

- Niveau 0 : rien comme le SOAP
  - Get et paramètres des URLs
- Niveau 1 : Utilisation des ressources
- Niveau 2 : Utilisation des verbes HTTP
- Niveau 3 : HATEOAS



Objets java ?

- Servlets / JSP

- Bean – *Plain Old Java Object* (POJO)
  - Constructeur par défaut
  - Propriétés avec getter / setter
  - Sérializable



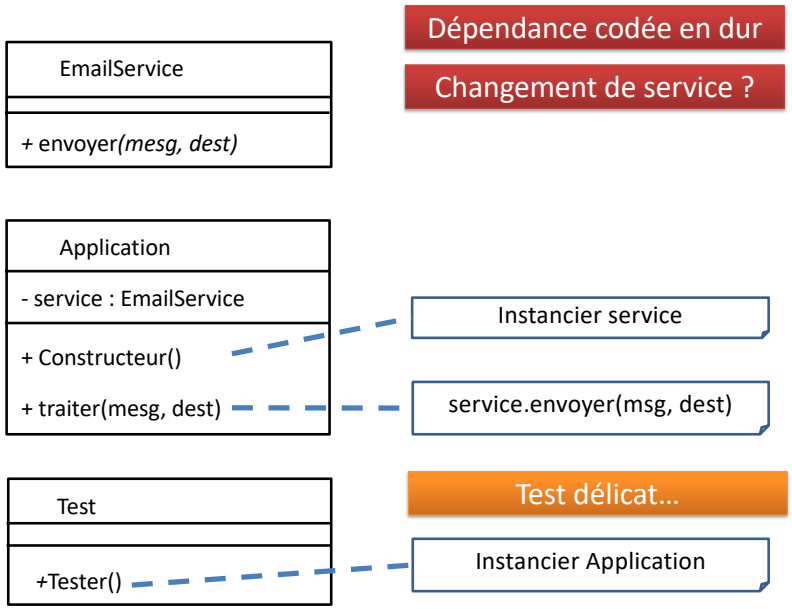
- Gestion de la vie de ces objets ?

Injection de dépendances

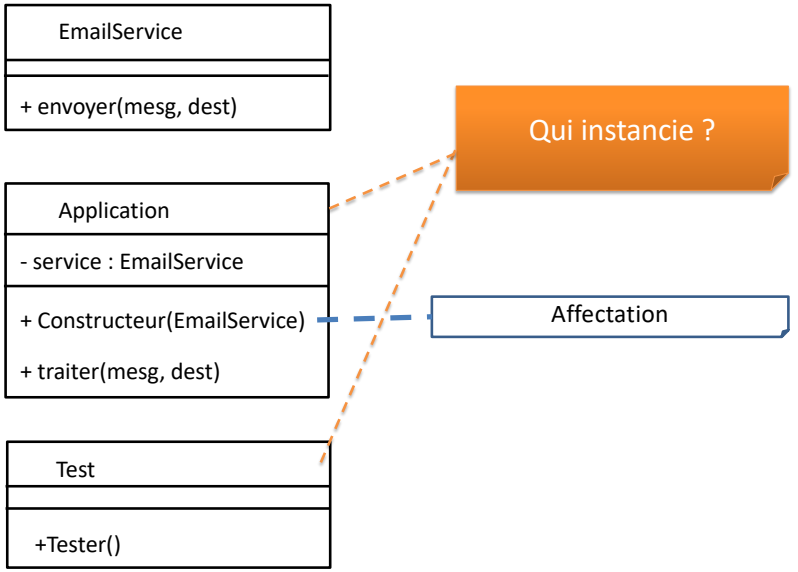
- Inversion de contrôle
- Limiter les dépendances (composition / héritage)
- Résoudre à ~~la compilation~~ l'exécution

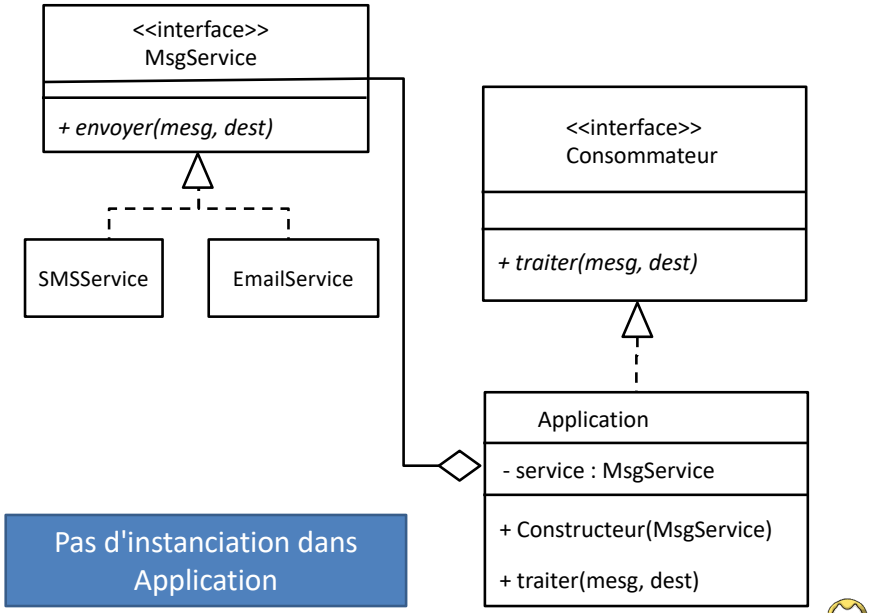
- Conteneur de dépendances ?
- Cycle de vie des objets

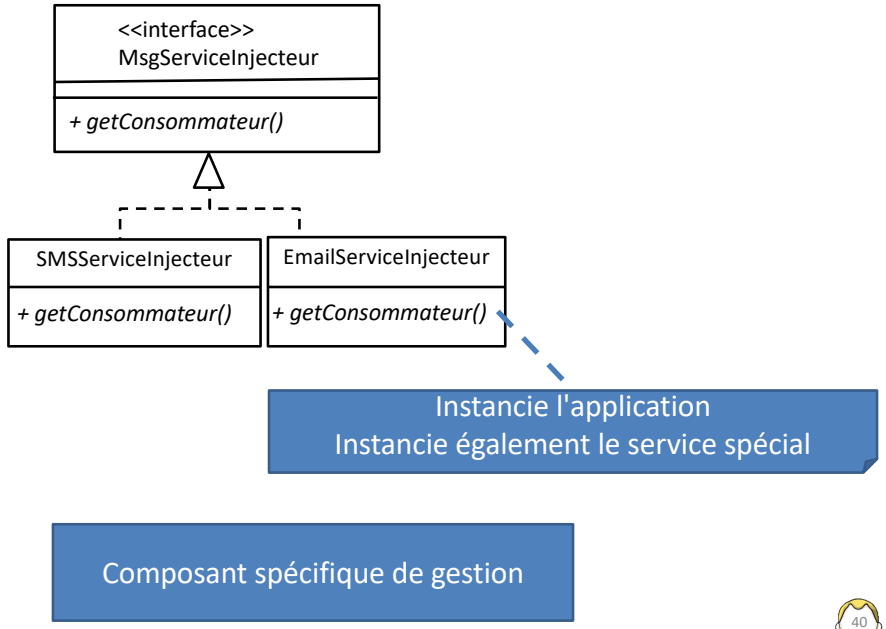
- Sympathique pour les tests
  - Mockups utilisables quand c'est compliqué



https://www.journaldev.com/2394/java-dependency-injection-design-pattern-example-tutorial







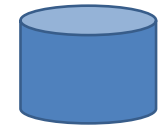
SPRING DATA

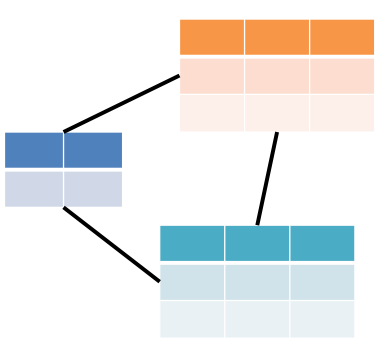




CLERMONT AUVERGNE INP Isima

Bases de données ?



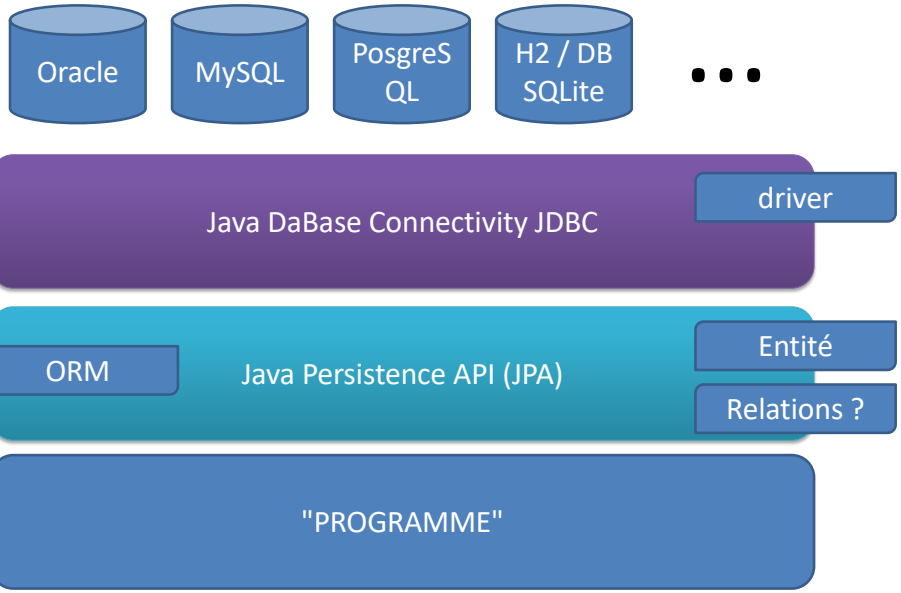


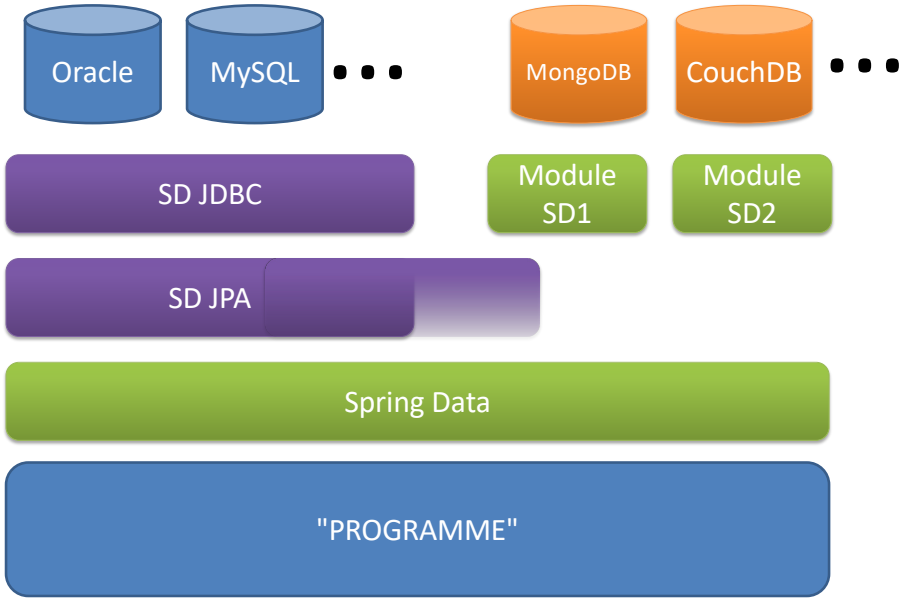
- Document
- Clé-valeurs
- Graphe
- Colonnes

Relationnel

SQL

NOSQL





Spring Data JPA

- Entité / Classe / Table
- Attribut / Colonne

- Repository / opérations sur la base
- Méthode / requête

Données en base…

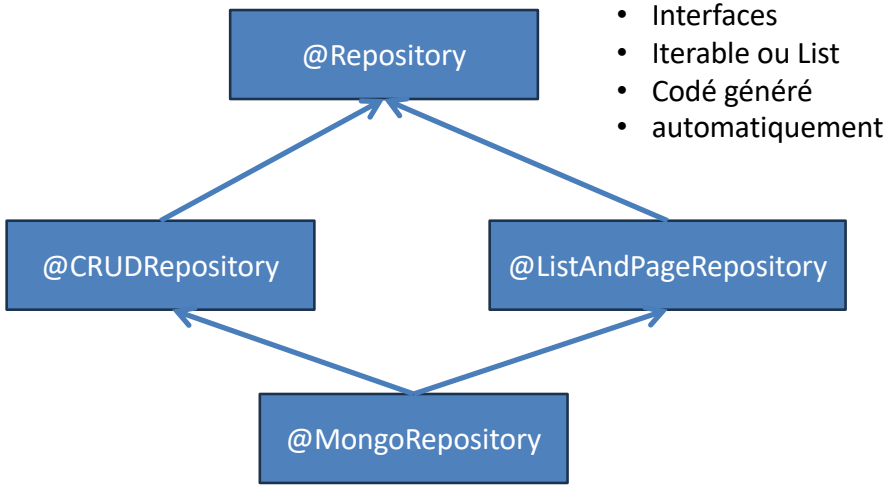
- Base relationnelles mais pas seulement
- ORM : Java Persistence API
  - Hibernate
  - EclipeLink
- Java DataBase Connectivity

  Abstraction de la base

  Connecteurs

  ~~SQLite~~

Repositories



- Interfaces
- Iterable ou List
- Codé généré
- automatiquement