# **Lexique :**

* Design Pattern : c'est un modèle de développement pour solutionner un problème (organiser le code) - Autre terme : Patron de conception
* REST : norme d'architecture basé sur Ajax/Json, architecture de communication entre client et serveur (qui sont indépendants ou découplés).
* SOAP : Par opposition au REST, le Front formule une requête, e Back lui restitue une page HTML (avec données). SOAP est employé en JSP et PHP
* IoC : Inversion Of Control ...
* 3-Tiers: un client, un serveur et un serveur de base de données (Tiers étant un acteur non physique durant l'exécution) - Client/Serveur correspondrait à 2 tiers.
* Thread :(processus légers ou tâche) ensemble d'instructions du langage machine d'un microprocesseur <http://dico.developpez.com/html/1469-Generalites-processus-leger.php>
* Multi-thread : Exécution de thread en parallèle
* Serialization: La sérialisation est un procédé introduit dans le JDK version 1.1 qui permet de rendre un objet ou un graphe d'objets de la JVM persistant pour stockage ou échange et vice versa. Cet objet est mis sous une forme sous laquelle il pourra être reconstitué à l'identique. Ainsi il pourra être stocké sur un disque dur ou transmis au travers d'un réseau pour le créer dans une autre JVM. C'est le procédé qui est utilisé, par exemple, par RMI. La sérialisation est aussi utilisée par les beans pour sauvegarder leurs états.
* API : Interface réutilisable dans un autre code de développement
* WEB SERVICE:Il s'agit d'une technologie permettant à des applications de dialoguer à distance via Internet, et ceci indépendamment des plates-formes et des langages sur lesquelles elles reposent. Pour ce faire, les services Web s'appuient sur un ensemble de protocoles Internet très répandus (XML, HTTP), afin de communiquer. Cette communication est basée sur le principe de demandes et réponses, effectuées avec des messages XML.
* MAVEN : Builder pouvant réaliser plusieurs tâches pour un code donné telles que le lancement, les tests, le nettoyage (clean), la compilation et le build et la gestion des dépendances
* JSON/XML/YAML/CSV : format de fichiers
* Hibernate/JPA : Hibernate est un outil de mapping objet/relationnel pour Java. C'est à dire qu'il permet de faire le lien entre la représentation objet des données et sa représentation relationnelle basée sur le schéma de la base de données. Hibernate est open-source et ne respecte pas les standards JDO et SDO. C'est un framework.<http://dico.developpez.com/html/3039-Langages-Hibernate.php>
* L'API JPA a tenu compte du succès d'Hibernate et en a repris la plupart des concepts. JPA est une spécification, une implémentation<http://tahe.developpez.com/tutoriels-cours/jpa/>
* IoC :
* CRUD : Create Read Update Delete
* Middleware
* Paradigme : Style ou orientation de programmation informatique (différents paradigme : POO, Programmation procédurale, programmation récursive, etc...)
* Design Pattern MVC
* Le modèle est la partie qui contient les données et les méthodes de traitement des données (la lecture, l’écriture, la suppression, etc…)
* La vue est la partie qui correspond à l’interface graphique en lien direct avec l’utilisateur. Elle n’effectue pas de traitements de données.
* Le contrôleur est la partie qui fait le lien entre le modèle et la vue et leur synchronisation.
* Interface
* Aggregation : Permet d'assembler des objets de base, afin de construire des objets plus complexes
* Framework
* Agile : Voir précédemment (agilité)
* ORM

### **Programmation Orientée Objet :**

* objet : caractérisé par son comportement et ses attributs
* prototypage :
* classe : un modèle permettant de représenter un objet réel
* instance
* méthode :
* héritage : une classe fille peut récupérer toutes les propriétés de la classe mère si elle en hérite.
* polymorphisme
* attribut : propriétés de la classe
* interface
* visiblité: private : dans la classe, public : partout, protected : dans le package ou héritage
* encapsulation : grâce à la visibilité, on peut masquer aux yeux d'un développeur extérieur tous les rouages d'un objet et donc l'ensemble des méthodes destinées à la gestion interne de l'objet, auxquelles le développeur final n'aura pas à avoir accès. Si on encapsule avec private, on utilise alors les getters/setters pour déterminer les droits de modification des autres développeurs.
* substitution
* abstraction
* Classe abstraite : classe qui ne peut être instanciée. Ex une classe chien peut être instanciée, une classe animal non. On peut associer des méthodes à l'animal (respirer, marcher, etc.)
* Méthode abstraite : avec une signature (un état un nom un paramètre) mais pas de contenu. La méthode qui héritera devra donc implémenter son contenu
* interface : une classe avec des méthodes 100% abstraite. Une interface peut hériter de deux classes : exemple chauve-souris qui hérite de la classe oiseau (parce quelle a des ailes) et de la classe mammifère.
* Final : pour une variable, une méthode ou une classe : constante; méthode non modifiable; classe non-modifiable : une classe final ne pourra pas contenir de polymorphisme
* static : élément qui appartient à la classe, une classe avec que des éléments static ne peut être instanciée
* Garbace Collector : ramasse-miettes, supprime les zones mémoire qui ne sont plus référençées, géré automatiquement en Java
* signature
* polymorphisme
* capacité d'un objet à modifier son comportement propre et celui de ses descendants au cours de l'exécution
* par redéfinition (override) : par redéfinition de la méthode avec même signature
* par surcharge (overload) : par ajout de paramètres
* par interface
* constructeur
* super : on fait appel à la classe mère
* this : on fait appel à l'objet
* persistance : Mécanisme qui s'assure que les échanges de données avec la base soient garanties sans risque de perte.
* itération
* collection
* révolution
* DOM (virtuel) : Ensemble d'objets et de comportements composant une page HTML (manipulable par jQuery par exemple). Le DOM virtuel est une interprétation d'un DOM fictif servant composer une page résultat permettant ainsi à certains langages de ne modifier qu'uniquement ce qui diffère avec le DOM normal (économie d'actions et voie royale vers le Single Page).
* JDBC

### **SQL**

### **JDBC**

### **Hors-programme :**

#### **Gestion de projet**

* Cycle en V / recette : est un modèle conceptuel de gestion de projet imaginé à la suite du problème de réactivité du modèle en cascade.
* Il permet, en cas d'anomalie, de limiter un retour aux étapes précédentes.
* Agilité : Projet découpé en plusieurs sous-cycles en V (entre autres)

\* procédure stockée : on met la procédure au niveau de la BDD, ensemble d'instructions SQL précompilées, stockées dans une base de données et exécutées sur demande par le SGBD qui manipule la base de données.<https://fr.wikipedia.org/wiki/Proc%C3%A9dure_stock%C3%A9e>

\* Trigger : déclencheur sur un ensemble d'actions (ex update ou insert)

Liens :

* <http://dico.developpez.com/html/3278-Business-Intelligence-DOLAP-Desktop-OLAP.php>
* <https://www.google.fr/imgres?imgurl=https://upload.wikimedia.org/wikipedia/commons/thumb/6/6a/Cycle_de_developpement_en_v.svg/300px-Cycle_de_developpement_en_v.svg.png&imgrefurl=https://fr.wikipedia.org/wiki/Cycle_en_V&h=153&w=300&tbnid=AZCvZ3ZkQxuaiM:&tbnh=107&tbnw=211&usg=__DgixHhhbaoGEeSzCEAG7f1DMHyI=&vet=10ahUKEwiauMjI8IDVAhWCXhoKHU6PAFYQ9QEIKzAA..i&docid=P4a0p_ny4-eQNM&sa=X&ved=0ahUKEwiauMjI8IDVAhWCXhoKHU6PAFYQ9QEIKzAA>
* <http://www.commentcamarche.net/contents/811-poo-le-polymorphisme>