Atelier Services Web

PLAN DU COURS

1) BASES DE DONNEES
2) WSDL & UUDI
3) WEB SERVICE SERVEUR ET CLIENT (SOAP)
4) WEB SERVICE SERVEUR ET CLIENT (REST)

Objectifs pédagogiques

Gérer des données

Développer des composants dans le langage d’une base de données

Utiliser un service Web (« web services »)

Intégrer le service Web à une application

Comprendre les problématiques modernes liées à la persistance des données

DEFINITION : La persistance des données consiste à **sauvegarder et restaurer** l'état des objets manipulés au sein des applications.

REMARQUE : Un des buts étant de prolonger la durée de vie des données **au-delà de la durée de vie d'une session** applicative.

Distinguer les différentes solutions de persistance de données

Le stockage des objets peut s'effectuer sur plusieurs types de supports, tels que XML, bases de données, fichiers binaires, fichiers plats… Les **SGBD relationnels** sont actuellement largement utilisés.

Le **système relationnel** offre de nombreux atouts tels que l'interopérabilité. Il offre de nombreuses **fonctionnalités sophistiquées**.

SGBD RELATIONNELS		
Performance	**Fiabilité / Sécurité**	**Intégrité**
Recherche	**Langage de requête**	**Indexation**
Relations	**Transactions**	**Concurrence**

SGBD relationnels et non relationnels

Les bases de données relationnelles ont de **nombreux avantages**, mais aussi des **inconvénients**.

Les bases de données relationnelles sont très utilisées en entreprise. La plupart des applications utilisent ce système.

Cependant, les système **NoSQL** (non relationnels) peuvent mieux convenir lorsque la quantité de donnée devient trop importante (MongoDB, Cassandra…).

En effet, **l'intégrité référentielle** des systèmes relationnels peut nuire aux performances.

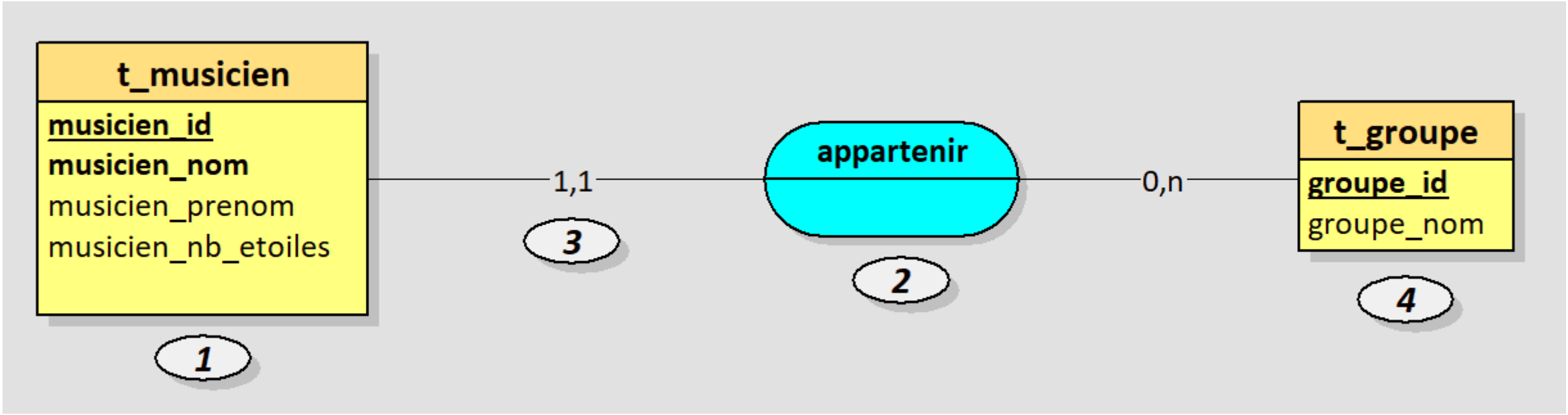
Également, **la redondance des données n'est pas souhaitable** dans les **SGBD relationnels**. C'est un principe à ne pas violer. On utilise même des algorithmes pour éliminer ces redondances dans la structure de la base de donnée. Alors que dans les **SGBD non relationnels**, **la redondance est tolérée** même lorsque la base est stockée sur plusieurs machines différentes.

Lecture et définition des cardinalités

Première Lecture en partant de **MUSICIEN**

➔ **Un musicien fait partie** de **1** et **1 seul groupe**

Algorithme de lecture
On doit positionner dans l'ordre prévu 1 => L'entité que nous analysons 2 => L'association étudiée 3 => La cardinalité de l'entité que nous analysons (MIN / MAX) 4 => L'entité de l'autre côté de l'association

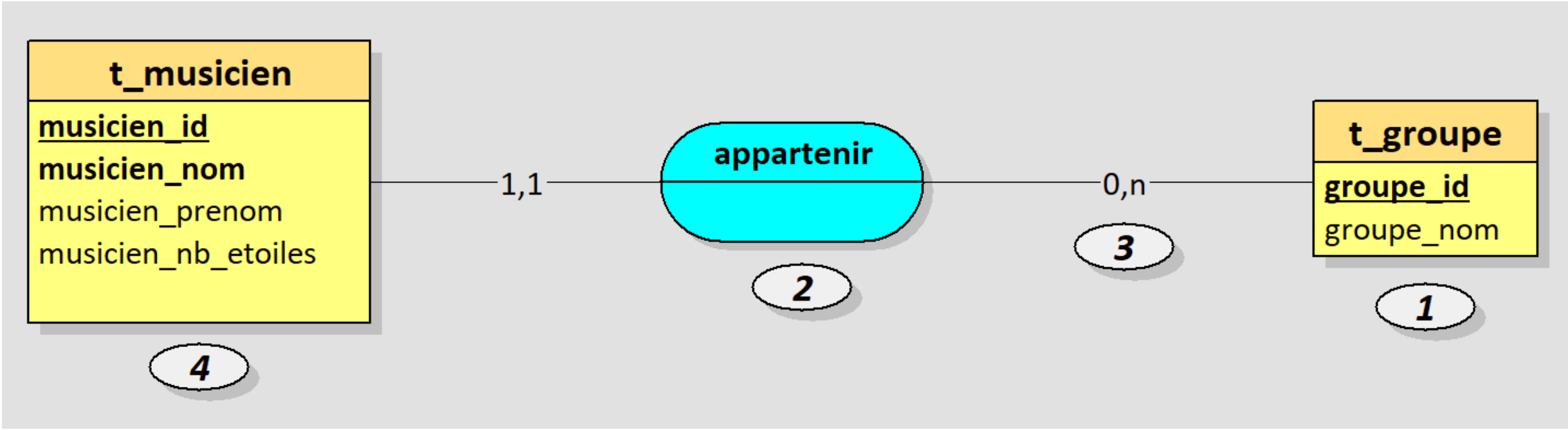


Lecture et définition des cardinalités

Deuxième lecture en partant de **GROUPE**

➔ **Un group contient 0 à n musiciens**

Algorithme de lecture
On doit positionner dans l'ordre prévu 1 => L'entité que nous analysons 2 => L'association étudiée 3 => La cardinalité de l'entité que nous analysons (MIN / MAX) 4 => L'entité de l'autre côté de l'association



Architecturer des composants d'accès aux données

Comment **modéliser les entités métier** en objets au sein de mes applications, et comment **assurer la persistance de ces objets de façon durable et fiable**, tout en bénéficiant de mécanismes efficaces pour les opérations de recherche et de mise à jour concurrente ?

Comment assurer **l'interopérabilité entre un existant relationnel** (le système d'information de l'entreprise) **et le modèle objet ?** Comment passer d'une représentation tabulaire (en SGBD) à une représentation sous forme de graphe (les objets) ?

NOTE : Il est primordial de simplifier la maintenance des applications et promouvoir la réutilisation des objets métier entre les applications.

TP1 : Préparation environnement JAVA

a) Télécharger le Driver JDBC correspondant à la base de donnée utilisée

b) Télécharger et installer le JDK

c) Télécharger et Installer votre environnement de travail (Eclipse Java EE)

d) Créer votre Workspace et configurer l'encodage et les fins de lignes

e) Configurer votre workspace, exporter le fichier de préférences

f) Créer un template pour vos commentaires (Window / Preferences / Java / Editor / Templates / New) (Exporter le template crée)

Config eclipse autocompletion

Window/Preferences/Java/Editor/ContentAssist/Advanced
Cocher/Décocher les éléments selon vos besoins

Déposer votre fichier « tp3_environnement-jdbc.zip » dans votre espace de dépôt de fichiers. Les mots de passe ne doivent pas figurer dans le fichier.

Exemple de configuration supplémentaire Eclipse permettant de montrer au développeur des erreurs supplémentaires
➔window/preferences/Java/Compiler/Errors-Warnings

Section Unnecessary code
Value of exception parameter is not used : ERROR
Unused import : ERROR
Unnecessary cast of « instanceof » operation : ERROR
Unnecessary declaration of thrown exception : ERROR
Unused « break » or « continue » label : ERROR

Section null analysis
Null pointer access : ERROR
Potential null pointer access : ERROR
Redundant null check : ERROR

EXEMPLE DE TEMPLATE A CRÉER :

```
// TRIGRAMME - ${currentDate:date('yyyy-MM-dd')}
// Params :
// Description :
```

TP2 : Conception MCD

a) Concevoir la base de donnée demandée

L'idée est de concevoir une véritable base de donnée répondant aux critères suivants :

- **Un musicien est associé à un et un seul slogan**
- **Chaque musicien ne joue que dans un seul groupe**
- **Les musiciens peuvent jouer avec plusieurs types d'instruments**

BONUS : Un musicien est associé à plusieurs concerts

Déposer votre fichier « tp2_conception-relationnelle.zip » dans votre espace de dépôt de fichier. Les mots de passe ne doivent pas figurer dans le fichier.

TP3 : Préparation environnement JAVA

a) Télécharger le Driver JDBC correspondant à la base de donnée utilisée

b) Télécharger et installer le JDK

c) Télécharger et Installer votre environnement de travail (Eclipse Java EE)

d) Créer votre Workspace et configurer l'encodage et les fins de lignes

e) Configurer votre workspace, exporter le fichier de préférences

f) Créer un template pour vos commentaires (Window / Preferences / Java / Editor / Templates / New) (Exporter le template crée)

Config eclipse autocompletion

Window/Preferences/Java/Editor/ContentAssist/Advanced
Cocher/Décocher les éléments selon vos besoins

Déposer votre fichier « tp3_environnement-jdbc.zip » dans votre espace de dépôt de fichiers. Les mots de passe ne doivent pas figurer dans le fichier.

Exemple de configuration supplémentaire Eclipse permettant de montrer au développeur des erreurs supplémentaires
➔window/preferences/Java/Compiler/Errors-Warnings

Section Unnecessary code
Value of exception parameter is not used : ERROR
Unused import : ERROR
Unnecessary cast of « instanceof » operation : ERROR
Unnecessary declaration of thrown exception : ERROR
Unused « break » or « continue » label : ERROR

Section null analysis
Null pointer access : ERROR
Potential null pointer access : ERROR
Redundant null check : ERROR

EXEMPLE DE TEMPLATE A CRÉER :

```
// TRIGRAMME - ${currentDate:date('yyyy-MM-dd')}
// Params :
// Description :
```

TP4 : Connexion JDBC à une base de donnée

a) Créer une documentation accompagnant le TP (vous la remplirez au fur et à mesure)
b) Créer une nouvelle base de donnée
c) Créer un nouveau projet « Tp_connexionJDBC » (Vérifier configuration du projet encodage et fin de lignes)
d) Ajouter le driver pour JDBC dans le projet (Java Build Path)
e) Réaliser la connexion à la base de donnée (Utiliser une sous classe, exemple : MyConnectionUtils)

L'idée est de réussir à se connecter à une base de donnée.

Si une erreur se produit, l'exception doit être est gérée et affichée.

Si la connexion est réussi, un message « Connexion OK » s'affiche

Déposer votre fichier « tp4_connexion-jdbc.zip » dans votre espace de dépôt de fichiers. Les mots de passe ne doivent pas figurer dans le fichier.

TP5 : Manipuler la base de donnée avec JDBC

a) Créer les tables prévues et des tuples dans la base de donnée.
b) Programmer en Java une requête simple pour récupérer les données

BONUS : Réaliser une requête update ou insert en utilisant les transactions

BONUS : Réaliser une requête avec un Prepared Statement.

Déposer votre fichier « tp5_acces-donnees.zip » dans votre espace de dépôt de fichiers. Les mots de passe ne doivent pas figurer dans le fichier.

TP6 : API REST et RESTFul

a) Rechercher des informations sur l'architecture REST de Roy Fielding

b) Ecrire un résumé des éléments que vous avez compris (1 page maximum papier scanné JPG/PNG ou PDF)

c) Comment le CRUD est inclut dans l'architecture proposée par Roy Fielding ?

d) Renseignez vous sur les différents LEVEL et les décrire

e) Qu'entend-t'on par la notion de VERB ?

f) Quel différence y-a-t'il entre REST et RESTFul

Déposer votre fichier dans votre espace de dépôt de fichiers. Les mots de passe ne doivent pas figurer dans le fichier. « tp_nom_prenom_architecture-rest.zip »

Web Services

Les Web services sont une composante importante dans la création des applications distribuées.

- SOAP (XML) – JAXWS (contient un spécial message XML)
- REST (JSON) – JAXRS (plus souple à implémenter)

On envoie une requête http à un serveur, qui traite la demande et répond. C'est ensuite au client qui a reçu cette réponse de reconstituer la page HTML.

Web Services

On utilise le protocole HTTP pour effectuer les communications

Les web services permettent aux applications d'échanger des données et d'effectuer des opérations indépendamment de leur plateforme d'exécution et du langage de programmation utilisé.

On peut considérer un web service comme étant une classe d'objet contenant des méthodes invocables à distance par des programmes clients.

L'idée est de pouvoir faire dialoguer les machines à travers le réseau.

Web Services

Exemple simplifié de REQUÊTE HTTP SOAP	**Exemple simplifié de REPONSE HTTP SOAP**
<?xml version="1.0" encoding="UTF-8"?> <Envelope> <Header/> <Body> <conversion> <montant>12</montant> </conversion> </Body> </Envelope>	<?xml version="1.0" encoding="UTF-8"?> <Envelope> <Body> <conversionResponse> <return>132</return> </conversionResponse> </Body> </Envelope>

Web Services

Trois concepts fondamentaux à connaître :

SOAP (Simple Object Access Protocol)

Est un protocole d'échange inter-applications indépendant de toute plate-forme, basé sur le langage XML

WSDL (Web Services Description Language)

Donne la description au format XML des Web Services en précisant les méthodes pouvant être invoquées, leur signatures, et le point d'accès (URL, port...)

UDDI (Universal Description, Discovery and Integration)

Normalise une solution d'annuaire distribué de Web Services, permettant à la fois la publication et l'exploration (recherche) de Web Services

NOTE : UDDI se comporte lui même comme un Web Service dont les méthodes sont appelées via le protocol SOAP

Qu’est ce que SOA ?

La SOA peut être alors vue telle une interface entre le fournisseur de service et le consommateur de service. Ce contrat de service pourra alors être implémenté dans les applications en Java, Microsoft.net ou encore d’autres langages de programmation.

WSDL : **W**eb **S**ervices **D**escription **L**anguage

NOTE : Le contrat de service est souvent définit à l'aide du Web Service Definition Language (WSDL) qui est une structure de balise standard basée sur XML.

REST

REST : **RE**presentational **S**tate **T**ransfer

Roy Fielding a mis en place en 2000 un style d'architecture pour la conception d'applications faiblement couplées sur HTTP. Il est souvent utilisé dans le développement de services Web.

REST ne définit aucune règle concernant la façon dont il doit être implémenté au niveau inférieur, il met simplement des directives de conception de haut niveau et nous laisse penser à notre propre implémentation.

Les API REST vous permettent de développer tout type d'application Web mettant en œuvre le CRUD (créer, récupérer, mettre à jour, supprimer).

NOTE : Les directives REST suggèrent d'utiliser une méthode HTTP spécifique sur un type particulier d'appel passé au serveur.

Webservice REST http Method and status Code

HTTP	CRUD	ALL - Ressource de collection (Ex : /musiciens)	ONE – Ressource unique (Ex : /musiciens/47)
GET	Read	**200 (OK),** liste des musiciens. Utilisez la pagination, le tri et le filtrage pour parcourir de grandes listes	**200 (OK)**, utilisateur unique. **404 (Not Found)**, si ID non trouvé ou invalide
POST	Create	**201 (Created)** en-tête « Location » avec un lien vers /users/{id} contenant un nouvel identifiant	Évitez d'utiliser POST sur une seule ressource
PUT	Update / Remplacement	**405 (Méthode non autorisée)**, à moins que vous ne vouliez mettre à jour toutes les ressources de l'ensemble de la collection de ressources	**200 (OK)** ou **204 (Pas de contenu)**. Utilisez 404 (Not Found), si l'ID n'est pas trouvé ou invalide
PATCH	Update partiel / Modification	**405 (Méthode non autorisée)**, sauf si vous souhaitez modifier la collection elle-même	**200 (OK)** ou **204 (Pas de contenu)**. Utilisez **404 (Not Found)**, si l'ID n'est pas trouvé ou invalide
DELETE	Delete	**405 (Méthode non autorisée)**, sauf si vous souhaitez supprimer toute la collection - à utiliser avec prudence	**200 (OK)**. **404 (Not Found)**, si ID non trouvé ou invalide

http status codes

Les **codes de statut** de réponse HTTP indiquent si une requête HTTP a été **exécutée avec succès ou non**. Le code HTTP permet de déterminer le résultat d'une requête ou d'indiquer une erreur au client.

NOTE : Ce code numérique est destiné aux traitements automatiques par les logiciels de client HTTP.

http Codes les plus courants (Voir RFC 26161 et RFC 72312)			
200 : succès de la requête	301 et 302 : redirection, permanente et temporaire	401 : utilisateur non authentifié	403 : accès refusé
404 : page non trouvée	405 : Non authorisée	500 et 503 : erreur serveur	504 : le serveur n'a pas répondu

UDDI

On utilise **l'annuaire UDDI** lorsque nous avons plusieurs entreprises ou plusieurs équipes « qui ne se connaissent pas » mais qui ont **besoin d'utiliser les web services**.

On va ainsi « **publier** » dans **l'annuaire UDDI le web service**. Le consommateur va pouvoir :

- « rechercher » le webservice
- prendre connaissance de ce web service
- et ainsi pouvoir « invoquer » et utiliser le webservice

JAX-WS

JAX-WS (anciennement appelé JAX-RPC) permet de développer très simplement des services Web en Java

JAX-WS fournit un ensemble d'annotations pour mapper la correspondance Java-WSDL. Il suffit pour cela d'annoter directement les classes Java qui vont représenter le service Web.

JAX-WS (SOAP)

Dans l'exemple ci-dessous, une classe Java utilise des annotations JAX-WS qui vont permettre par la suite de générer le document WSDL. Le document WSDL est auto-généré par le serveur d'application au moment du déploiement

NOTE : operationName permet de choisir le nom de la méthode, si non spécifié, c'est le nom de la méthode qui sera utilisé
NOTE : webParam permet de choisir le nom du paramètre
NOTE : Autre exemple d'annotation : @XmlTransient

```
@WebService(serviceName="BanqueWS")
public class BanqueService { … }

@WebMethod(operationName="Conversion")
public double conversion(@WebParam(name="montant")double mt) {return mt*100;}
```

```
// Publication du Web Service sur le Serveur
Endpoint.publish(url, new BanqueService());
```

```
// Utilisation du Webservice par le Client
BanqueWS stub = new BanqueServiceService().getBanqueWSPort();
double res = stub.conversion(100);
```

JAX-RS (REST)

Dans l'exemple ci-dessous, on implémente la spécification REST à l'aide d'annotations (GET et POST)

NOTE : Nous avons ici un large ensemble d'automatismes, dont la récupération des instances et la transformation en JSON

Autre exemple d'annotation : @JsonIgnore

```
@GET
@Path("{id}")
@Produces(MediaType.APPLICATION_JSON)
public Response get(@PathParam("id") long i_long_id)
{
…
}
```

@GET
(Récupérer)

```
@POST
@Consumes(MediaType.APPLICATION_JSON)
@Produces(MediaType.APPLICATION_JSON)
public Response createAccount(Montant montant) {
…
}
```

@POST
(Créer)

JAX-RS (REST)

Dans l'exemple ci-dessous, on implémente la spécification REST à l'aide d'annotations (PUT et DELETE)

NOTE : Nous avons ici un large ensemble d'automatismes, dont la récupération des instances et la transformation en JSON

```
@PUT
@Path("{id}")
@Consumes(MediaType.APPLICATION_JSON)
public Response update(
        @PathParam("id") long i_long_id,
        Montant montant
) {
…
}
```

@PUT
(Modifier)

```
@DELETE
@Path("{id}")
public Response delete(@PathParam("id") long
i_long_id) {
…
}
```

@DELETE
(Supprimer)

Activités de développement professionnel

Nécessité d’effectuer une veille permanente, continue et réitérée

Effectuer un stage professionnel permettant vraiment de vous perfectionner

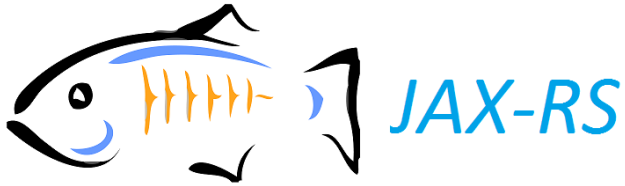
Communiquer, échanger avec vos collègues, documenter

Effectuer des concours

Logiciels et systèmes

Logiciels et systèmes

POSTMAN



Bibliographie et liens

[1] Les Compagnons du DevOps, "Qu'est-ce qu'un Ingénieur DevOps, un SysOps ou AdminSys ? - En Solo #11", https://youtu.be/M_a5J0csyD8

[2] Sergio, "Create Mock from Wsdl with SoapUI", https://youtu.be/VwTOFsY4cWQ

[3] qualite logiciel, "Les WEB SERVICES", https://youtu.be/Rbmzac50kBQ

[4] coursera, "4.2.4 - Publication et découverte de services (UDDI)", https://www.coursera.org/lecture/service-oriented-architecture/4-2-4-service-publication-and-discovery-uddi-ejXxi

[5] Java Brains, "SOAP Web Service Basics", https://www.youtube.com/playlist?list=PLqq-6Pq4lTTZTYpk_1DOowOGWJMIH5T39

[6] LaMalditaProgramadora, "SOAP Web Service + Spring Boot 3 (3.0.6) + JPA [JDK 17]", https://youtu.be/S5hBIHxfaHk

[7] Professeur Mohamed YOUSSFI, "Web services", https://www.youtube.com/playlist?list=PLxr551TUsmAozms7qX1iT8JzAwllHq0vD

[8] Baeldung, "Introduction to JAX-WS", https://www.baeldung.com/jax-ws

[9] Baeldung, "Generate WSDL Stubs with Maven", https://www.baeldung.com/maven-wsdl-stubs

[10] Code Java, "Java RESTful Web Services CRUD API Examples with Jersey and Tomcat", https://youtu.be/-_VPzhKJPfE

[11] Oracle, "Java Downloads", https://www.oracle.com/java/technologies/downloads

Bibliographie et liens

[12] ANSSI, "Les règles de sécurité", https://www.ssi.gouv.fr/administration/protection-des-oiv/les-regles-de-securite/

[13] Eclipse foundation, "Eclipse IDE Download", https://www.eclipse.org/downloads/packages/

[14] Jakarta EE, "An open, community-driven platform for future innovation of enterprise Java technologies", https://jakarta.ee/release/

[15] DevStory, "Tutoriel Java JDBC", https://devstory.net/10167/java-jdbc

[16] Mysql, "MySQL Community Downloads", https://dev.mysql.com/downloads/mysql/

[17] HeidiSQL, "What's this?", https://www.heidisql.com/

[18] Looping, "Modélisation Conceptuelle de Données", https://www.looping-mcd.fr/

[19] ZDNet, "Panorama des solutions de persistance objet en architecture .NET", https://www.zdnet.fr/actualites/panorama-des-solutions-de-persistance-objet-en-architecture-net-39145424.htm

[20] Raphaël Fournier-S'niehotta, Philippe Rigaux, "Applications orientées données", http://orm.bdpedia.fr/

[21] ObjectDB, "JPA Criteria API Queries", https://www.objectdb.com/java/jpa/query/criteria

[22] MariaDB, "MariaDB Downloads", https://mariadb.com/downloads/

Bibliographie et liens

[23] Dominique Liard, "Java Database Connectivity", https://koor.fr/Java/JDBC.wp

[24] Dominique Liard, "Introduction de l'utilisation de JPA", https://koor.fr/Java/TutorialJEE/jee_jpa_introduction.wp

[25] Matthew Tyson, "What is JPA? Introduction to the Java Persistence API", https://www.infoworld.com/article/3379043/what-is-jpa-introduction-to-the-java-persistence-api.html

[26] Java Community Process, "JSR 338: JavaTM Persistence 2.2", https://jcp.org/en/jsr/detail?id=338

[27] Java Performance Tunning, "Java Performance Tuning", https://www.javaperformancetuning.com/news/interview041.shtml

[28] Hibernate, "Hibernate OGM - Your NoSQL datastores. One API.", https://hibernate.org/ogm/

[29] Victoria Farber, "Hibernate validation in a standalone implementation", https://www.infoworld.com/article/2137346/java-tip-hibernate-validation-in-a-standalone-implementation.html

[30] Apache JDO, "Why JDO ?", https://db.apache.org/jdo/why_jdo.html

[31] Wikipedia, "Mapping objet-relationnel", https://fr.wikipedia.org/wiki/Mapping_objet-relationnel

[32] SQL.sh, "Cours", https://sql.sh/cours

Bibliographie et liens

[33] Laurent Audibert, "Bases de données et langage SQL", https://laurent-audibert.developpez.com/Cours-BD/?page=conception-des-bases-de-donnees-modele-a

[34] Frédéric BAURAND, "06 - Merise - MCD - Lecture association", https://youtu.be/2Oskf-eiB60

[35] ZetCode, "MySQL Java", https://zetcode.com/db/mysqljava/

[36] Mocodo online, "MCD Online generator", http://www.mocodo.net/

[37] DBMS Tools, "Data modeling tools", https://dbmstools.com/categories/data-modeling-tools

[38] Insum, "SQL Developer Data Modeler Just what you need", https://youtu.be/NfrUy-TYP_8

[39] REST API Tutorial, "Create REST APIs with JAX-RS After learning to ", https://restfulapi.net/create-rest-apis-with-jax-rs/

[40] MariaDB, "Setting Character Sets and Collations", https://mariadb.com/kb/en/setting-character-sets-and-collations/

[41] Eclipse Foundation, "Eclipse IDE for Enterprise Java and Web Developers", https://www.eclipse.org/downloads/packages/release/2024-03/r/eclipse-ide-enterprise-java-and-web-developers

[42] Apache Tomcat, "Apache Tomcat", https://tomcat.apache.org

[43] SOAP UI, "Download the Best API Testing Tool for your Needs", https://www.soapui.org/downloads/soapui/

[44] Advanced REST Client, "An API consumption tool for developers", https://install.advancedrestclient.com/