

Jax-Y


Déployez vos Web services REST Sans code

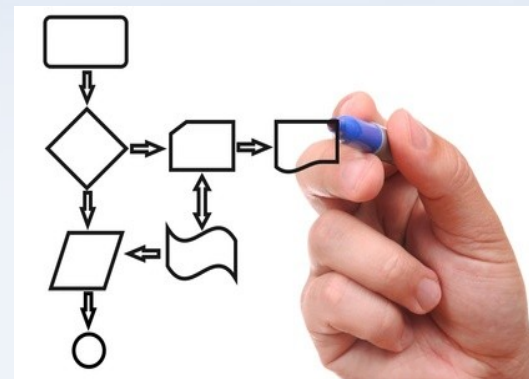


R. Yahiaoui

Agenda




Technique 



Conception 



Démo 



Code 



Petit retour d'expérience

Jax-Y

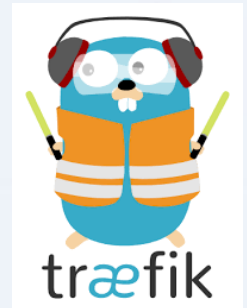
- Pourquoi
- Quels usages
- Comment
 - Serveur d'App
 - War vs Jar
 - Runtime Compiler

REST

- Définition
- Architecture
- Desing API
- Critères qualité

Démo

@bout me : **R. Yahiaoui**

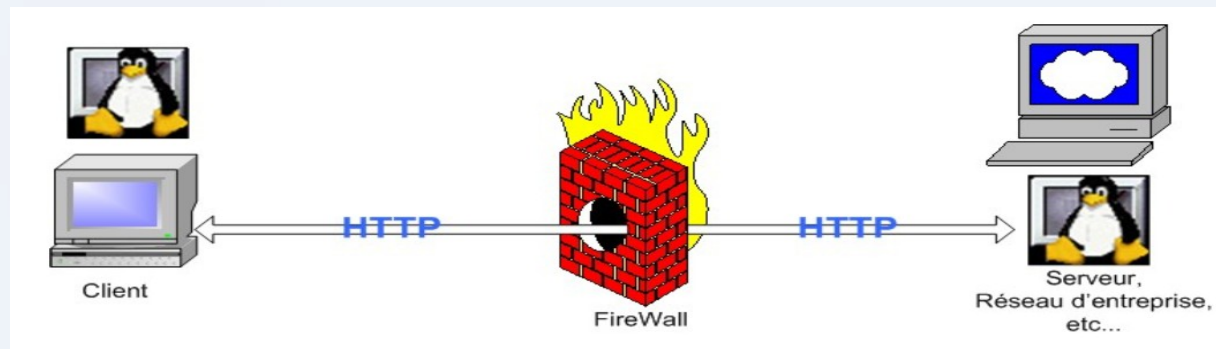




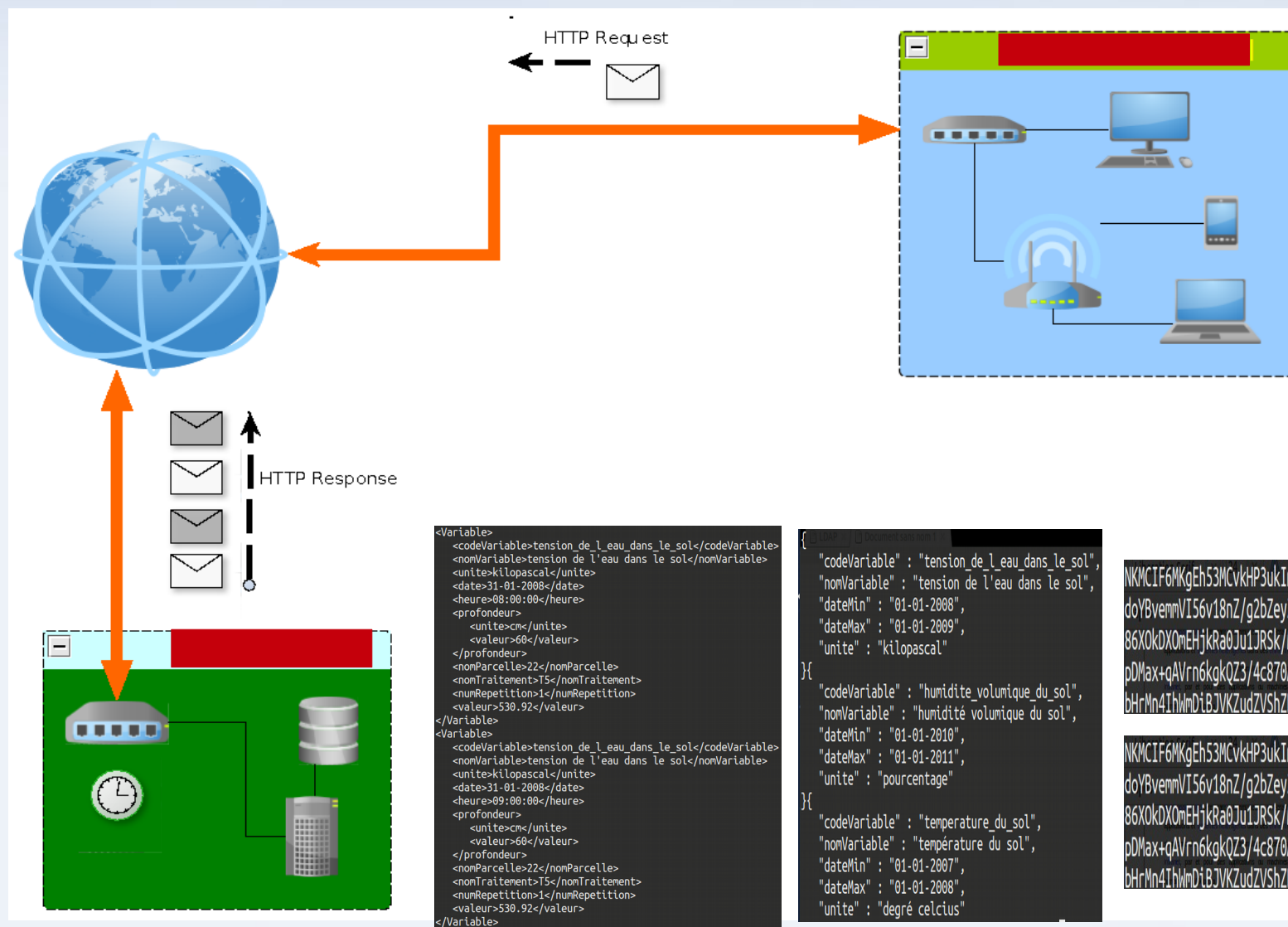


Web-Service

Un service web est un **programme informatique** de la famille des **technologies web** permettant la communication et l'échange de données entre applications et **systèmes hétérogènes** dans des **environnements distribués**. Il s'agit donc d'un ensemble de fonctionnalités exposées sur **internet** ou sur un **intranet**, par et pour des applications ou machines, sans intervention humaine... (Wikipedia)



Retour d'expérience (2014)



NKMCIF6MKgEh53MCvkHP3ukImURFiYV8mMR/575tt0XGeE
doYBvemvVI56v18nZ/g2bZeyZ0baXmLEKYf4zmh7d/ioIH
86XOkDXOmEHjkRa0Ju1JRSk/Mm9+KxoVZQ45ss9cFMTLSk
pDMax+qAVrn6kgkQZ3/4c870AkF015ocDZ8httXjYwki4t
bHrMn4IhWmDiBJVKZudZVShZB1NVEF4Wct0L7p5ZnvMnwfi

NKMCIF6MKgEh53MCvkHP3ukImURFiYV8mMR/575tt0XGeE
doYBvemvVI56v18nZ/g2bZeyZ0baXmLEKYf4zmh7d/ioIH
86XOkDXOmEHjkRa0Ju1JRSk/Mm9+KxoVZQ45ss9cFMTLSk
pDMax+qAVrn6kgkQZ3/4c870AkF015ocDZ8httXjYwki4t
bHrMn4IhWmDiBJVKZudZVShZB1NVEF4Wct0L7p5ZnvMnwfi



Ouvrir accès à la base de données ?

Oui mais ...

- SQL, peut être Complexe & Dangereux ?!
- Mise à jour des Schémas
- Forte charge sur la BDD vs caching
- Scalabilité
- Media-Type (DB- Relationnelles)
- Nouvelles architectures web
- Friendly DB access



Quelle architecture ?

SOAP (SOA)	REST (ROA)
<p>Standard (Protocole générique).</p> <p>Verbeux (XML, SEI)</p> <p>Contrats formels</p> <p>Fort Couplage Client – Serveur</p> <p>Opérations avec ou sans état (stateless)</p> <p>Couche transport générique : Http , smtp, jms...</p>	<p>Opérations sans état (stateless)</p> <p>Protocole HTTP pour le transport</p> <p>Accessible, Facile à mettre en œuvre</p> <p>Utilise les verbes standards du protocole HTTP pour faire du CRUD (GET PUT POST DELETE)</p>



Representational State Transfert

Propriétés architecturale

- * Performances
- * Scalabilité
- * Simplicité
- * Visibilité
- *

Contraintes architecturale

- * Client-server
- * Stateless
- * Cacheable
- * Uniform interface



REST — Uniform interface

Identification des **ressources**

Resource as URIs
<http://my-api.com/v1/cars/123>

Manipulation des ressources
grâce à des **représentations**

JSON, XML

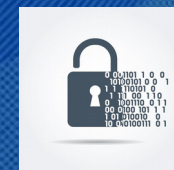
Action sur les ressources
(Self-descriptif messages)

HTTP GET, PUT, POST, DELETE

HATEOAS

(Hypermedia As The
Engine Of Application State)

Hypermedia APIs
JSON-LD



REST — HTTP methods

	http://my-api.com/v2/cars/	http://my-api.com/v2/cars/123
GET	Renvoie la liste de toutes les voitures	Renvoie la voiture ayant l'identifiant 123
POST	Crée une nouvelle voiture	Erreur
PUT	Remplace la collection par une nouvelle collection de voitures	Met à jour la voiture ayant l'identifiant 123
DELETE	Supprime toutes les voitures	Supprime la voiture ayant l'identifiant 123



REST — Règles de nommage

- * Noms ou verbes .../my_cars/21 (.../logout)
- * Camel case .../myCars/21
 - UpperCamelCase .../MyCars/21
 - LowerCamelCase .../myCars/21
 - Snake_case .../my_cars/21
 - Dashed-snake-case .../my-cars/21
- * Singulier / Pluriel

Ce n'est qu'une question de **goût...**

Faire un **choix** et s'y tenir



REST — Pagination (Query parameters)

- Page number : **?page=81**
- Page size : **?size=10**
- Curseur (Insertion-Friendly) : **?cursor=69ae13**
- Sort

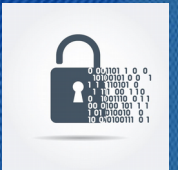




REST — HTTP status codes

- **1xx** : Hold on
- **2xx** : Here you go !
- **3xx** : Go away
- **4xx** : You fucked up :-)
- **5xx** : I fucked up :-(





Critères de Qualité



Flexibilité



Robustesse



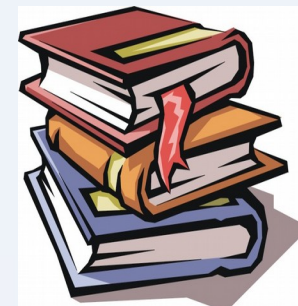
Performance



Réutilisabilité



Sécurité

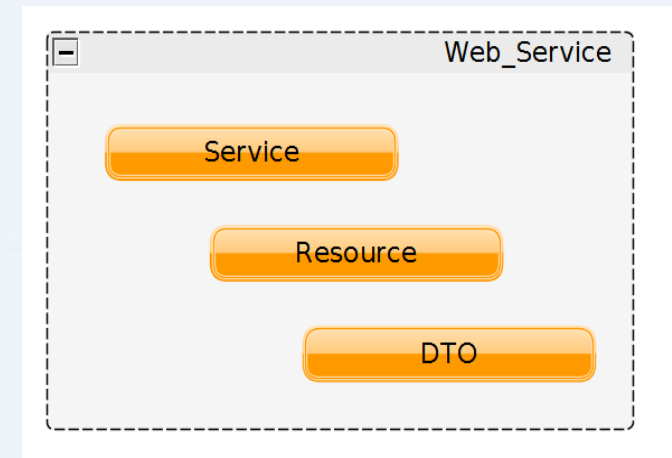
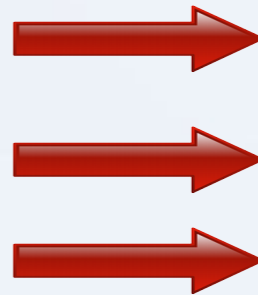


Documentation



Flexibilité

Un web service est une « *Fonction* » capable d' **intercepter** des requêtes HTTP, d'en extraire la demande, de l'**exécuter**, puis **formater** le résultat pour enfin le renvoyer au client.



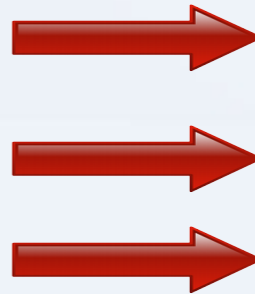
Package service web



Performances (Temps de réponse)

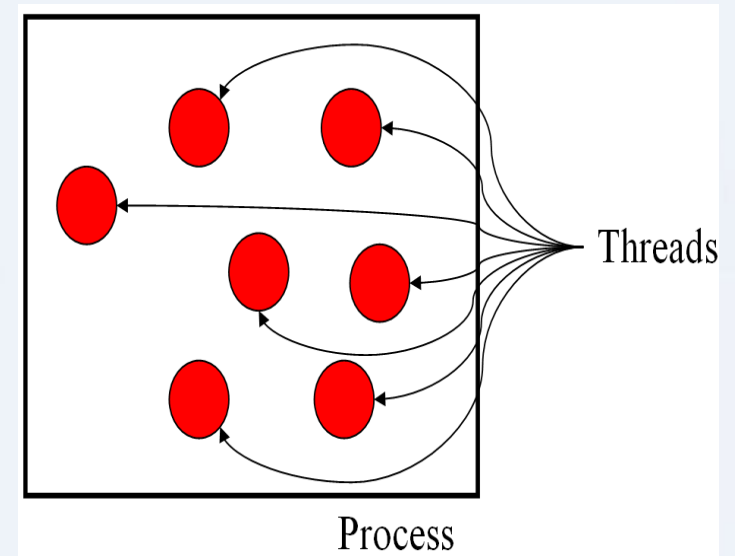
Une thread (appelée aussi processus léger) est un fil d'instructions (un chemin d'exécution) à l'intérieur d'un processus.

Contrairement aux processus, les threads d'un même processus partagent le même espace d'adressage et le même environnement..



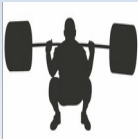
~~Dénormalisation DB~~

VS



Débit d'extraction SANS // ~ 0,5 Mo/s

Débit d'extraction AVEC // ~ 9 Mo/s

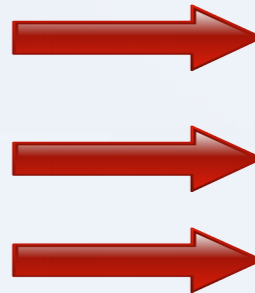


Robustesse

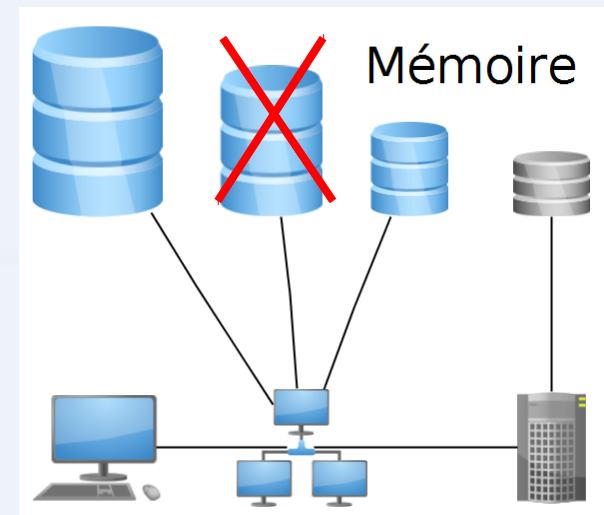
Contrôler la charge mémoire
(empreinte mémoire) afin d'éviter
toute défaillance du système.

Pagination des requêtes SQL

LIMIT – OFFSET



Volume de données >> à la mémoire

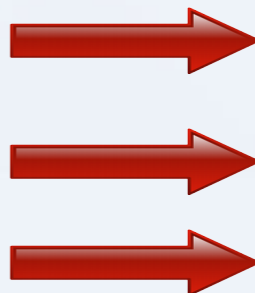




Réutilisabilité

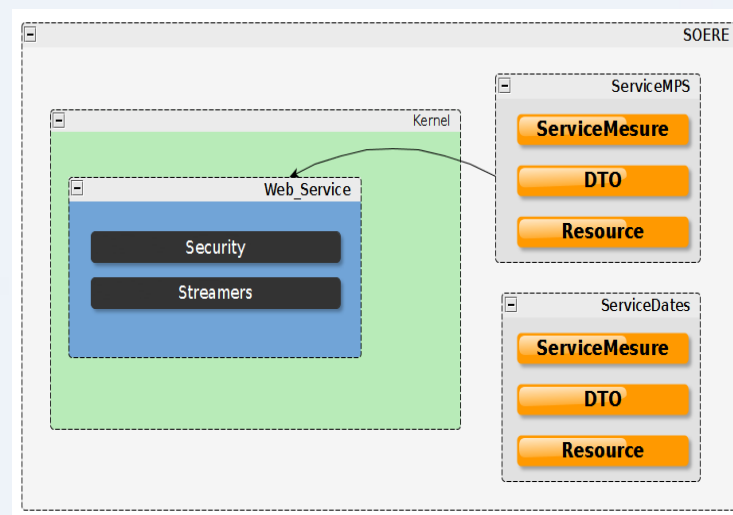
Intégration du web service (core)
dans le Kernel du S.I .

⇒ Héritage implicite des
fonctionnalités du web service.



Libre à chaque S.I d'implémenter ses
propres web services
(dans la couche plugin)

Write once, run anywhere



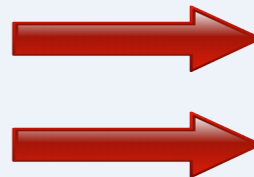


Documentation

Spécification pour documenter les API Rest
(Via annotation)



Problème : Pas de prise en
charge des sub-resource
locator



Propre API qui repose
sur l'introspection du
Code (sans annotation)

Info : Nouvelle API Swagger : Sans annotation ni configuration particulière

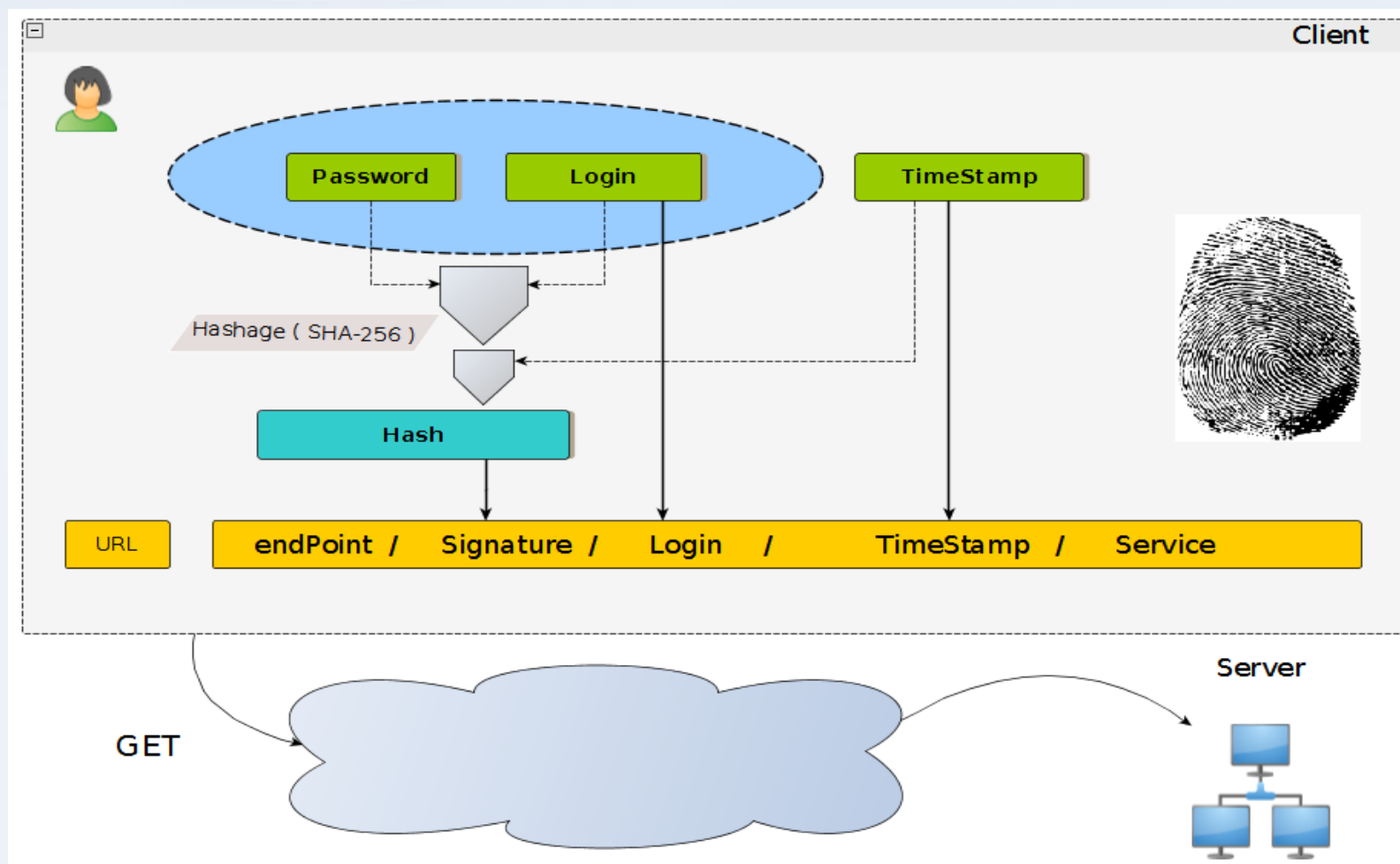
<https://github.com/sdaschner/jaxrs-analyzer>



Sécurité (1/2)

~~SSL~~

Client

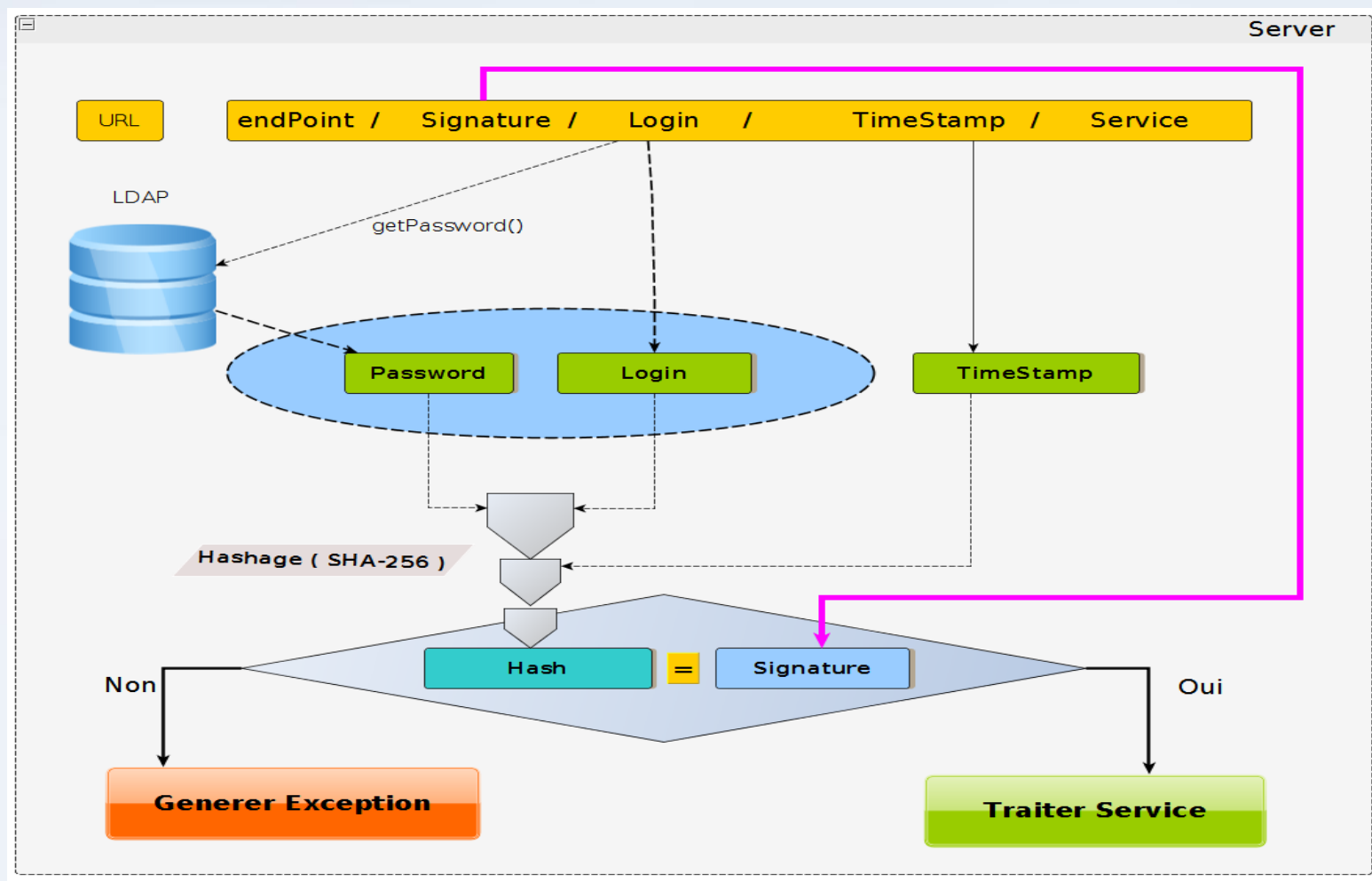


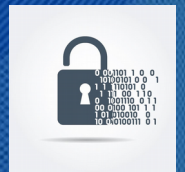


Sécurité (2/2)

~~SSL~~

Serveur





Sécurité (Alternative)

Keycloak

JWT

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJhdXRoLWJhY2t1bmQiLCJzdiI0Ijoc2FibG9ubmllcmUiLCJhdWQiOiJ3ZWItZnJvbnRlbmQiLCJleHAiOjE0NTg2Mzg2MTgsIm5iZiI6bnVsbCwiaWF0IjoxNDU4NjM1MDE4LCJqdGkiOiI4NWEzY2I0Yy01NzIwLTRkYmEtYWU5NC05MzFkNjA5MzdjNDciLCJ1YW1lIjoiaSHViZXJ0IFNBQkxPTk5Jw4hSRSI0InBlcm1pc3Npb25zIjpbIkVESVRPUiIsIlJFVklFV0VSIl19.fwxQ6GrPRyEi8wKh1BpYrWPuiF0pPswSahgM1WFWIx0
```




Sécurité (Alternative)

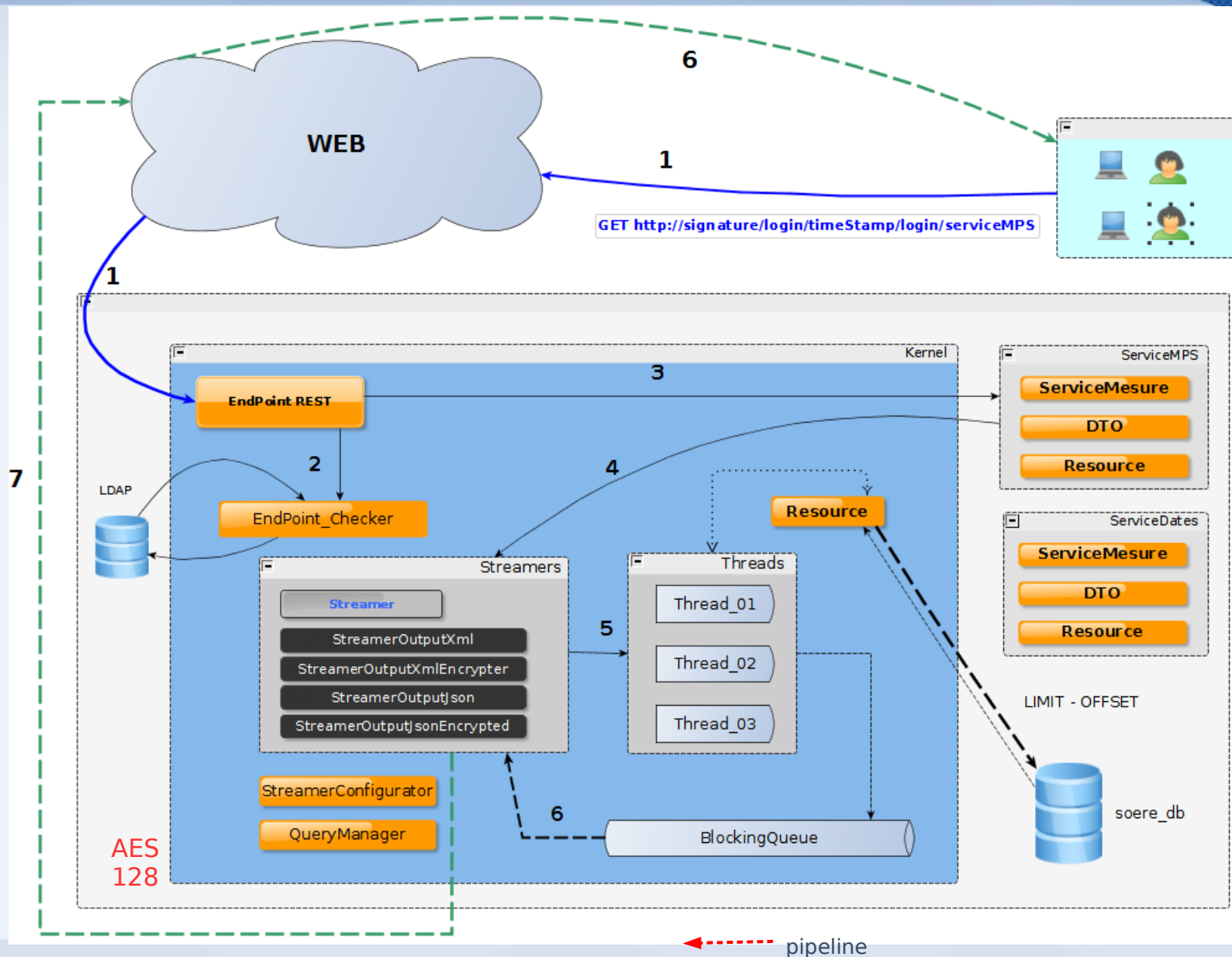
JWT

```
> base64URLdecode(encodedHeader);  
'{"alg":"HS256","typ":"JWT"}'
```

```
> base64URLdecode(encodedPayload);  
'{"iss":"auth-backend","sub":"hsablonniere","aud":"web-frontend",  
"exp":1458638618,"nbf":null,"iat":1458635018,"jti":  
"85a3cb4c-5720-4dba-ae94-931d60937c47","name":"Hubert  
SABLONNIÈRE","permissions":["EDITOR","REVIEWER"]}'
```

```
> signature  
fwxQ6GrPRyEi8wKh1BpYrWPuiF0pPswSahgM1WFWIxo
```

Fonctionnement du système





Jax-Y (2017)

→ **Open source Project**



→ **Liens :**

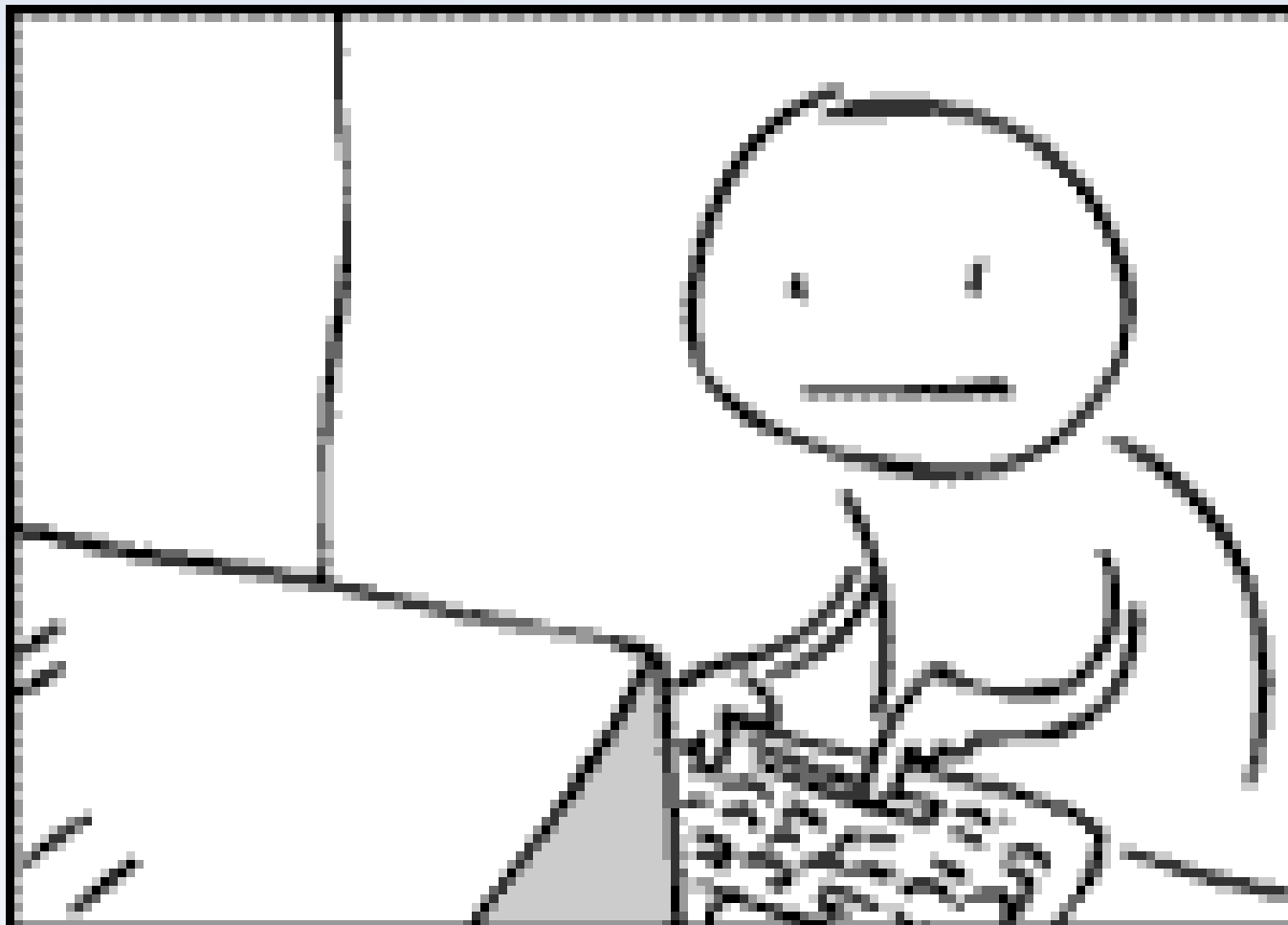
<https://github.com/rac021/Jax-Y>

https://github.com/rac021/Jax-Y/tree/master/demo_sourceForge

→ **SourceForge (Jax-Y + Client-UI)**

<https://sourceforge.net/projects/jax-y/>

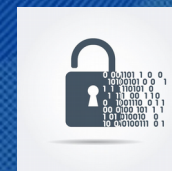
Happy Dev - Happy users



Code.. from scratch !

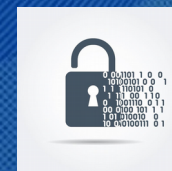
Happy Dev - Happy users





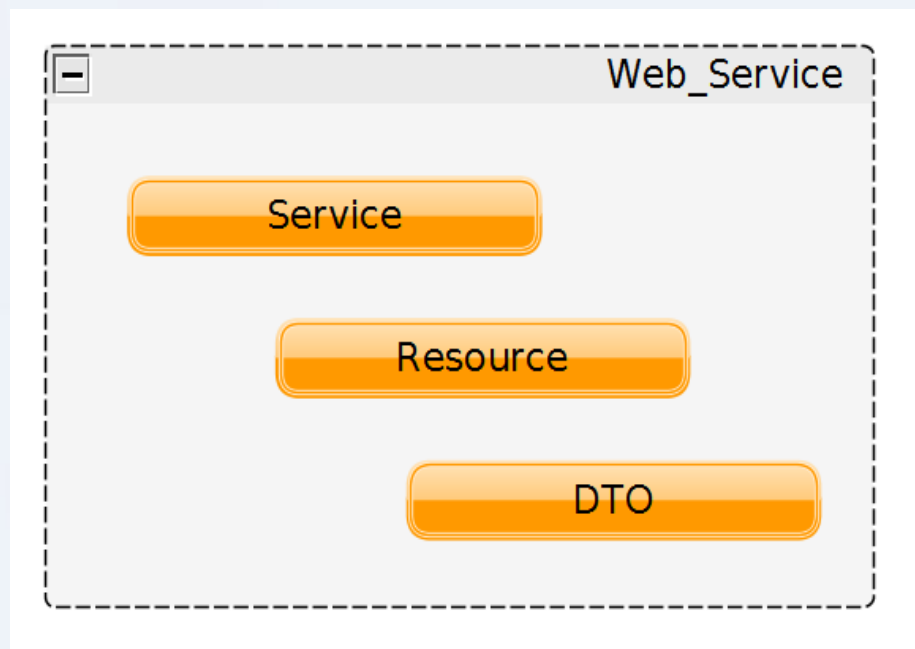
Jax-Y, Use Cases :

- Approche « **Database** as 'secured' web service »
- Déployer un web service avec « **zéro** » compétence informatique (ou presque)
- Faire **interagir** 2 (ou plusieurs) S.I
- Utilisation comme brique (applicative) dans une **architecture micro services**



Comment ?

Runtime Compilation. Why not ?



Package service web

Renseigner des
templates en
fonction des
différentes
configurations au
run

Construire et
compiler les
services
au déploiement de
l'application



Java-EE - App Server

JAX-WS	JSF	EJB	JMS
Interceptors	CDI	JAX-RS	Batch
WebSocket	JPA	JSON-P	JCA
JSP	JSTL	JTA	JavaMail
Servlets	Expression Language	Bean Validation	JASPIC
			Concurrency

- Taille de plusieurs centaines de Mo
- Permet de déployer des WARs (de petites tailles en général)
- Demande certaines compétences pour son administration



WAR vs JAR

WAR ? yes , but ...



- Gros serveur
- déploiement rapide
- Nécessite de l'admin

JAR ? yes , but ...



- Jar « customisé »
- Long à déployer
- Admin pas nécessaire

"Make JAR, not WAR".

So ?



- Spring boot
- Micro services & Cloud



Archi

G-Jax-Api

G-Jax-Service-Discovery

G-Jax-Security-Provider

Jax-Y

G-Jax-Client



Démo





Upcoming Features :

- Swagger-Angular-Client intergration
- Runtime Algo Choice for Encryption (AES - DES ...) (Done !)
- Global Configuration Supports (Thread pool size, nb of threads service , data queue size) (Done !)
- Authentication server using HTTPS (Done !)
- GUI Https supports (Done !)
- Generate Jar Client for specific Configuration
- Add Real Time decryption (For Stream + Large data)
- Add support Let's Encrypt

MERCI DE VOTRE
ATTENTION

Question ?