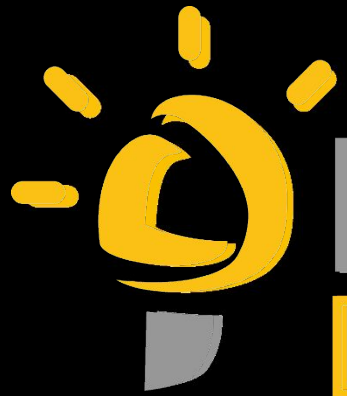

Innovativ Days 2018

Smarter Together

GraphQL :
Créez votre API data-driven

Innovativ Days
Smarter Together

Présentation de GraphQL

Innovativ Days
Smarter Together

GraphQL / C'est quoi ?

- Langage de requétage typé
- Créé à l'origine par Facebook
- Utilisé pour créer des APIs

Innovativ Days
Smarter Together

A quoi ça ressemble ?

- Requete / Réponse
- Query / Mutation / Subscription
- Exemple : « Shema 1 »

Innovativ Days
Smarter Together

Souscription : désolé pas de démo

Requete :

```
subscription {
  machineMetric(machineId:"123456-456789", metric:"currentPower") {
    value
    at
  }
}
```

Stream de réponse

```
{
  "data": {
    "machineMetric": {
      "value": 128,
      "at": "27/10/2018-14:47:38"
    }
  }
}
```

Innovativ Days
Smarter Together

“Graph“ dans GraphQL ?

- Accéder aux même donnée depuis des chemins différents
- Exemple : « Shema 2 »

Innovativ Days
Smarter Together

GraphQL / Rest ?

- **Avantages de GraphQL**
- Validation des types
- Seulement les infos nécessaires
- Une seule requête
- Préférence pour GraphQL lors du développement de l'API.
- Plus naturel (REST : simple CRUD)

Innovativ Days
Smarter Together

GraphQL / Rest ?

- **Inconvénients de GraphQL**
- Types recursifs
- Nombre de requêtes internes au backend plus nombreuses (solution par la mise en place de cache)

Innovativ Days
Smarter Together

Concept :
API orientée service ou API orientée données

Innovativ Days
Smarter Together

Vision service / donnée

- WebService, EJB : services
- GraphQL, REST: données.

Innovativ Days
Smarter Together

Vision service / donnée

- Chaque type de donnée comporte ses propres comportements

- Exemple : « Schema 7 »

Innovativ Days
Smarter Together

Vision service / donnée

- Permissions : appliquées sur les données, non plus sur l'accés à un service.

Innovativ Days
Smarter Together

API orientée données : point de vue de l'architecture

Innovativ Days
Smarter Together

Architecture micro-service

- Le code métier ressemble à des « pipes » d'appels de micro-service
- GraphQL est très compatible avec ce genre d'architecture
- Abstrait la complexité d'une architecture orientée service en api orienté données

Innovativ Days
Smarter Together

Architecture hétérogène

- La plus répendue
- Abstrait la complexité d'une architecture hétérogène.
- A chaque nœud de la requête : un service différent peut répondre (DB, RMI)

A chaque noeud sa solution

```
{
  allMachines {
    type
    id
    serialNumber
    description {
      description
      power
    }
    status
  }
}

subscription {
  machineMetric(machineId:"123456-456789", metric:"currentPower") {
    value
    at
  }
}
```

DB MySQL de machines installées
- id, type, serial number

API de Smartpart pour “Description des machines”

API de Timelab pour le status tream temps réel

Innovativ Days
Smarter Together

Vision orientée donnée : point de vue développeur

Innovativ Days
Smarter Together

Dév de l'API

- Exemple : « schema 4, 5, 6 et 7 »
- La conception est naturelle (un peu comme les langages objets)
- La structure du code : « micro-data »
- Dans un « micro-data » tous les parents sont valides

Innovativ Days
Smarter Together

Utilisateur de l'API

- GraphiQL
- UI : Exemple ReactJS

- Une UI n'affiche pas des services mais affiche et réagit à des données.

Innovativ Days
Smarter Together

Conclusion

Innovativ Days
Smarter Together

Conslusion

- GraphQL reste un langage d'API.
- Ne remplace pas les services, GraphQL les utilisent.
- Une manière naturelle d'exposer les données d'un SI.

Innovativ Days
Smarter Together

Merci de votre attention !