Design et conception des Web API Rest

Ressources

- Doc PDF sur le design des API REST (en)
- Bonnes pratiques pragmatiques pour les API REST (en)
- Yahoo Elide
- JSON API
- JSON Schema
- Wordpress REST API
- API for business opportunities
- Slideshare Resful API Design (2ème édition)
- 10 bonnes pratiques pour les API REST (en)
- Théorie REST et HATEOAS (fr)
- API-Platform : solution PHP pour construire une API REST

Documentation d'API REST

- Voir le mémo sur la doc des API REST

CORS et services REST

- CORS et REST
- Explication sur CORS
- Utilisation de htaccess et PHP pour CORS

HATEOAS

HATEOAS : fait de rajouter des liens vers les ressources et actions possibles dans les réponses des web services. La façon de renvoyer ces liens n'est pas standardisées. Les propositions actuelles :

- HAL,
- JSON-LD, Hydra,
- JSON:API,
- JSON Schema

Mise en cache

- Gestion de l'entête Cache-Control
- Entête HTTP utilisés dans le cadre de la gestion du cache côté navigateur
- Présentation des entêtes HTTP de gestion du cache

Création, modification et suppression multiples (bulk operation)

D'une façon générale, comme pour GET utiliser le filtrage à l'aide de la *query string* sur l'url indiquant l'ensemble des ressources. Exemples :

- **[GET] /api/v1/users?id=1,2,3** : permet de récupérer seulement les utilisateurs dont l'id vaut 1, 2 ou 3.
- **[POST] /api/v1/users** : passer dans le corps de la requête un tableau d'objets à ajouter. Le service web devra détecter la présence d'un objet simple ou d'un tableau d'objet pour savoir s'il doit ajouter un ou plusieurs éléments. Le statut de la réponse devrait être 303 et elle devrait contenir des liens vers l'ensemble des ressources créées.
- **[PATCH] /api/v1/users?id=1,2,3** : permet de mettre à jour partiellement les utilisateurs d'id 1, 2 et 3. Le corps de la requête contient le champ à mettre à jour pour ces 3 utilisateurs.
- **[PUT] /api/v1/users?id=1,2,3** : permet de mettre à jour les utilisateurs d'id 1, 2 et 3. . Le corps de la requête contient un objet avec les données à mettre à jour pour ces 3 utilisateurs.
- **[DELETE] /api/v1/users?id=1,2,3** : permet de supprimer les utilisateurs dont l'id vaut 1, 2 ou 3.

Ressources :

- Solution de "RESTful Webservices Cookbook" (O'Reilly)
- Changement partiellement une collection avec PUT ou DELETE
- REST : supprimer une collection d'objets
- Supprimer plusieurs enregistrements avec REST

Authification et REST

Dans chaque web service suivant (hormis pour le POST), le token est transmis dans un header (*Authorisation: Bearer <token>*). Normalement, l'ensemble des accès aux web services doivent se faire en HTTPS pour éviter le vol de token (*Man in the middle*).

- **[GET] /authentication?verify=true** : pour vérifier le token courant (L'utilisateur existe-t-il toujours ? Le token est-il toujours valide ? Le token n'a-t-il pas été blacklisté ?).
- **[GET] /authentication?refresh=true** : pour rafraichir le token courant. Vérifie la validité du token et le met à jour la date de validité du token.
- **[POST] /authentication** : pour s'identifier. Identifiant (courriel) et mot de passe doivent être transmis dans le corps de la requête. Le mot de passe peut être transmis sous forme de hash.
- **[DELETE] /authentication** : : pour se désidentifier. Une réponse 204 doit être renvoyé si tout est ok.

Ressources :

- Retour d'expérience

Les verbes

Pour certains types de ressources, il peut être utile d'utiliser des verbes :

- calculate
- convert

- search
- count