Spécifications des webservices Parasites

Codes de réponses HTTP :

Code	Description
200 ok	L'opération s'est correctement déroulée
400 bad request	La requête est malformée. Un des champs est invalide.
403 forbidden	Vous ne possédez pas les droits pour une telle opération
404 not found	Ressource non trouvée
500 Error	Le serveur a rencontré une erreur

URL de base :

Le serveur n'étant pour le moment qu'en local, l'url à utiliser est :

localhost :3000/

(Vérifier donc que ce port soir non utilisé sur son ordinateur)

Généralités :

- Lors d'un appel au webservice du jeu Parasites, toutes les réponses seront envoyées au format JSON.
- La réponse comportera toujours un champs **result** qui vaudra 1 (réussite) ou 0 (échec).

Principe de Tokens :

Pour plus de sécurité, et mis à part la connexion d'un utilisateur et sa création, l'ensemble des appels aux différentes rouotes devront comprendre un Token. Celui-ci vous sera donné lors de la connexion d'un utilisateur, dans le champs **« token »**.

Ce token permet de s'assurer que l'utilisateur est bien connecté avant de lui fournir des données considérées senseibles.

Création d'un utilisateur :

URL :
http://localhost:3000/users/create

TYPE :
HTTP POST

PARAMETRES :

Nom	Type	Description	Requis ?
Aucun	JSON	Informations utilisateur	Oui

Exemple paramètre :

```
{
        "lastname": "solde",
        "firstname": "robin",
        "pseudo": "leekwars",
        "password": "bu78ZArd58",
        "email": "robin@hotmail.fr",
        "phone": "0606060606"
}
```

Vérifications paramètres :

- Lastname et Firstname ne doivent être composés que de lettres
- L'email doit être un email valide
- Le mot de passe doit être composé d'au moins 2 majuscules, 2 minuscules et 2 chiffres
- Le numéro de téléphone ne doit être composé que de chiffres

Connexion d'un utilisateur :

URL :

http://localhost:3000/users/connect

TYPE :

HTTP POST

PARAMETRES :

Nom	Type	Description	Requis ?
Aucun	JSON	Pseudo et mot de passe	Oui

Exemple paramètre :

```
{
        "pseudo": "leekwars",
        "password": "bu78ZArd58"
}
```

Vérifications paramètres :

- Le pseudo et le mot de passe doivent correspondre à un utilisateur dans la Base de données

Valeurs de retour :

- Lorsque les informations sont correctes, l'API renvoie un message de succès, le token de l'utilisateur, et la durée de vie du token :

```
{
 "result": 1,
 "message": "Athentification is a success",
 "expires in": "1 day",
 "token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOjMsImV4cCI6MTQ5MTczODcyNjgzN30.3z
9DHdcUJ2kYqGsxISoYRrBNjqt7MDWmeIMRJLIgRwc"
}
```

- Lorsque les informations ne sont pas correcte :

```
{
 "result": 0,
 "message": "Athentification failed"
}
```

Récupérer tous les utilisateurs :

URL :

http://localhost:3000/users

TYPE :

HTTP GET

PARAMETRES :

Nom	Type	Description	Requis ?
token	string	Token de connexion	Oui

Exemple paramètre :

?token=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOjMsImV4cCI6MTQ5MTczODcyNjgzN30.3z9DHdcUJ2kYqGsxISoYRrBNjqt7MDWmeIMRJLIgRwc

Vérifications paramètres :

- Le token doit être valide

Valeurs de retour :

- Token expiré:

```
{
 "result": 0,
 "message": "Acces token has expired"
}
```

- Token valide :

```
[
 {
  "id": 1,
  "pseudo": "toto"
 },
 {
  "id": 2,
  "pseudo": "drakorn"
 }
]
```

Récupérer un utilisateur :

URL :

http://localhost:3000/users/<user_id>/

TYPE :

HTTP GET

PARAMETRES :

Nom	Type	Description	Requis ?
token	string	Token de connexion	Oui

Exemple paramètre :

?token=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOjMsImV4cCI6MTQ5MTczODcyNjgzN30.3z9DHdcUJ2kYqGsxlSoYRrBNjqt7MDWmeIMRJLlgRwc

Vérifications paramètres :

- Le token doit être valide
- L'id doit être correcte

Valeurs de retour :

- Id invalide :

```
{
 "result": 0,
 "message": "No user found"
}
```

- Id valide :

```
{
 "id": 3,
 "email": "robin@hotmail.fr",
 "lastname": "solde",
 "firstname": "robin",
 "pseudo": "leekwars",
 "password": "bu78ZArd58",
 "phone_number": "0606060606",
 "created_at": "2017-04-02T13:44:01.000Z",
 "updated_at": "2017-04-02T13:44:01.000Z",
 "deleted_at": null
}
```

Mettre à jour des données (le squelette est le même pour tous les champs):

URL :

http://localhost:3000/users/update/<bdd_field>/<new_value>/

TYPE :

HTTP PUT

BDD fields :

Vous pouvez, de la même manière, changer les champs suivants :

- pseudo
- email
- firstname
- lastname
- password

Il vous suffit de placer dans l'URL le mot clé correspondant, au niveau du <bdd_field>.

PARAMETRES :

Nom	Type	Description	Requis ?
token	string	Token de connexion	Oui

Exemple paramètre :

?token=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOjMsImV4cCI6MTQ5MTczODcyNjgzN30.3z9DHdcUJ2kYqGsxISoYRrBNjqt7MDWmeIMRJLIgRwc

Vérifications paramètres :

- Le token doit être valide
- La <new_value> doit correspondre aux tests de validités cités plus haut (création d'utilisateur)

Valeurs de retour :

- Value incorrecte, exemple avec le password :

```
{
 "result": 0,
 "errors": [
  "Validation error: Your password must contain 2 Uppercase characters, 2 minor cases and 2 digits"
 ]
}
```

- Value correcte, exemple avec le password :

```
{
 "result": 1,
 "message": "Password correctly updated"
}
```

Ajouter un ami

URL :

http://localhost:3000/users/friends/add/<friend_id>

TYPE :

HTTP POST

PARAMETRES :

Nom	Type	Description	Requis ?
token	string	Token de connexion	Oui

Exemple paramètre :

?token=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOjMsImV4cCI6MTQ5MTczODcyNjgzN30.3z9DHdcUJ2kYqGsxlSoYRrBNjqt7MDWmeIMRJLlgRwc

Vérifications paramètres :

- Le token doit être valide
- L'id de l'autre utilisateur doit être correcte et différend du sien

Valeurs de retour :

- ID incorrecte :

```
{
 "result": 0,
 "message": "You try to add as a friend a user that does not exist"
}
```

- ID correcte :

```
{
 "result": 1,
 "message": "Friend correctly added"
}
```

Lister les amis d'un utilisateur

URL :

http://localhost:3000/users/friends/all

TYPE :

HTTP POST

PARAMETRES :

Nom	Type	Description	Requis ?
token	string	Token de connexion	Oui

Exemple paramètre :

?token=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOjMsImV4cCI6MTQ5MTczODcyNjgzN30.3z9DHdcUJ2kYqGsxISoYRrBNjqt7MDWmeIMRJLIgRwc

Vérifications paramètres :

- Le token doit être valide
- Attention, ne sont compris comme « amis » que les utilisateurs ayant eux-même ajoutés l'utilisateur comme amis

Valeurs de retour :

- Présence d'amis (affiche un tableau d'ID des amis) :

```
[
 1
]
```

Supprimer un ami

URL :

http://localhost:3000/users/friends/remove/<friend_id>

TYPE :

HTTP DELETE

PARAMETRES :

Nom	Type	Description	Requis ?
token	string	Token de connexion	Oui

Exemple paramètre :

?token=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOjMsImV4cCI6MTQ5MTczODcyNjgzN30.3z9DHdcUJ2kYqGsxISoYRrBNjqt7MDWmeIMRJLIgRwc

Vérifications paramètres :

- Le token doit être valide
- L'id de l'autre utilisateur doit être correcte et différend du sien
- L'autre utilisateur doit être un ami (les deux doivent s'être mutuellement ajouté)

Valeurs de retour :

- ID incorrecte ou amitié non acceptée :

```
{
"result": 0,
"message": "Users are not friends"
}
```

- ID correcte et amitié valide:

```
{
 "result": 1,
 "message": "Friendship correctly removed"
}
```