MOVING WebServices POST

*Spécifications*

Ce document synthétise les spécifications logicielles relatives aux WebServices en écriture qui devront être exposés par le BackOffice Moving-BO

Benjamin Verney – Thomas Badin

06/05/2014

MOVING WebServices POST

Spécifications

Table des matières

[Introduction 0](#_Toc387842231)

[Contrôle d’accès 0](#_Toc387842232)

[Cas standard 0](#_Toc387842233)

[Cas Limites 0](#_Toc387842234)

[Cas d’erreurs 0](#_Toc387842235)

[Accès aux éléments 0](#_Toc387842236)

[Méthode getListAllParcours 0](#_Toc387842237)

[Cas standard 0](#_Toc387842238)

[Cas Limites 0](#_Toc387842239)

[Cas d’erreurs 0](#_Toc387842240)

[Méthode getParcoursArchitectureById 0](#_Toc387842241)

[Cas Standard 0](#_Toc387842242)

[Cas Limites 0](#_Toc387842243)

[Cas d’erreurs 0](#_Toc387842244)

[Méthode getParcoursById 0](#_Toc387842245)

[Cas standard 0](#_Toc387842246)

[Cas Limites 0](#_Toc387842247)

[Cas d’erreurs 0](#_Toc387842248)

[Méthode getSousParcoursById 0](#_Toc387842249)

[Cas standard 0](#_Toc387842250)

[Cas Limites 0](#_Toc387842251)

[Cas d’erreurs 0](#_Toc387842252)

[Méthode getTransitionById 0](#_Toc387842253)

[Cas standard 0](#_Toc387842254)

[Cas d’erreurs 0](#_Toc387842255)

[Méthode getSceneById 0](#_Toc387842256)

[Cas standard 0](#_Toc387842257)

[Cas d’erreurs 0](#_Toc387842258)

[Méthode getMediaById 0](#_Toc387842259)

[Cas standard 0](#_Toc387842260)

[Cas Limites 0](#_Toc387842261)

[Cas d’erreurs 0](#_Toc387842262)

[Méthode getArtefactById 0](#_Toc387842263)

[Cas standard 0](#_Toc387842264)

[Cas d’erreurs 0](#_Toc387842265)

[Méthode getPointdInteretByTag 0](#_Toc387842266)

[Cas standard 0](#_Toc387842267)

[Cas d’erreurs 0](#_Toc387842268)

[Méthode getPointdInteretListByGPSRange 0](#_Toc387842269)

[Cas standard 0](#_Toc387842270)

[Cas limite 0](#_Toc387842271)

[Cas d’erreurs 0](#_Toc387842272)

[Description des services (WSDL) 0](#_Toc387842273)

# Introduction

Les web services permettent d’enregistrer des objets métiers de l’extérieur dans le BackOffice. Les principaux objets métiers sont les parcours et sous parcours, les scènes, et les éléments (médias ou artefact) liés aux scènes.

Les WebServices seront utilisés à la fois par le Front Office Web et les applications mobiles :



Ce document ne traite que des WebServices permettant la contribution.

Les WebServices seront en [SOAP](http://fr.wikipedia.org/wiki/SOAP), protocole le plus répandu actuellement. Le Framework PHP utilisé pour le back office ([Zend Framework 2](http://framework.zend.com)), permet directement d’exposer des services (cf [Zend/Soap/Server](http://framework.zend.com/manual/2.0/en/modules/zend.soap.server.html) ). C’est donc ce que nous utiliserons pour créer les WebServices.

Dans l’ensemble de ce document, nous allons spécifier la réponse du WebService par un exemple. Ce sera moins formel qu’une définition des réponses au format XSD, mais plus simple à comprendre.

# Authentification

Nous utiliserons le système d’identification et autorisation OAuth2 décrit dans le fichier de spécifications « BackOffice-WebservicesReadOnly »

Chacune des requêtes webservices POST prendra en paramètre un « access token » obtenu par le système d’authentification. Ce token est lié au compte utilisateur et permet d’identifier ce dernier.

L’utilisateur devra disposer des droits de contribution pour pouvoir utiliser les méthodes de publication décrites plus loin, de plus son identifiant sera associé à la contribution.

# Validation des éléments

Les éléments envoyés par les contributeurs seront dans un premier temps à l’état de brouillon.

L’action d’un administrateur sera nécessaire pour valider l’entrée du contributeur et la passer dans l’état « public ».

# Méthodes de contribution des éléments

## Méthode PostMedia

Cette méthode permet d’envoyer un nouveau média au BackOffice. Elle prend en paramètre un objet média, l’access token (obligatoire) et un paramètre optionnel : l’ID de la scène associée au média.

### Cas Standard

L’utilisateur dispose des droits de contribution, l’access token est valide, les données du média sont complètes et l’ID de la scène (si fourni) correspond à une scène existante.

Le retour est alors de ce type :

<result>  
 <success value=’true’ />  
 <message value=’Media enregistre avec succes’ />  
 <media id="255"/>  
</result>

### Cas Limites

Dans le cas d’une erreur connue, le message de la réponse de la méthode sera adapté.  
On aura par exemple un retour du type :

Si le média est incomplet :

<result>  
 <success value=’false’ />  
 <message value=’Error: Le Media ne contient aucune data’ />  
 <media id="null"/>  
</result>

Si l’access token est invalide.

<result>  
 <success value=’false’ />  
 <message value=’Error: Invalid Token’ />  
 <media id="null"/>  
</result>

Si la scène parente précisée n’existe pas :

<result>  
 <success value=’false’ />  
 <message value=’Error: Scene inconnue ‘/>  
 <media id="null"/>  
</result>

### 

Si l’user n’a pas les droits :

<result>  
 <success value=’false’ />  
 <message value=’Error: Scene inconnue ‘/>  
 <media id="null"/>  
</result>

### Cas d’erreurs

Si un des paramètres obligatoires n’est pas fourni ou si

# Description des services (WSDL)

Une description des WebServices au format WSDL devra être générée automatiquement. Il servira de contrat sur l’utilisation des services entre le client (les APIs), et le serveur. Il servira aussi à générer plus rapidement les différentes APIs.