

IBM Campaign  
Version 9.1  
25 octobre 2013

## *Spécifications API*



**Important**

Avant d'utiliser le présent document et le produit associé, prenez connaissance des informations générales figurant à la section «Remarques», à la page 63.

LE PRESENT DOCUMENT EST LIVRE EN L'ETAT SANS AUCUNE GARANTIE EXPLICITE OU IMPLICITE. IBM DECLINE NOTAMMENT TOUTE RESPONSABILITE RELATIVE A CES INFORMATIONS EN CAS DE CONTREFACON AINSI QU'EN CAS DE DEFAUT D'APTITUDE A L'EXECUTION D'UN TRAVAIL DONNE.

Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. Les informations qui y sont fournies sont susceptibles d'être modifiées avant que les produits décrits ne deviennent eux-mêmes disponibles. En outre, il peut contenir des informations ou des références concernant certains produits, logiciels ou services non annoncés dans ce pays. Cela ne signifie cependant pas qu'ils y seront annoncés.

Pour plus de détails, pour toute demande d'ordre technique, ou pour obtenir des exemplaires de documents IBM, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial.

Vous pouvez également consulter les serveurs Internet suivants :

- <http://www.fr.ibm.com> (serveur IBM en France)
- <http://www.can.ibm.com> (serveur IBM au Canada)
- <http://www.ibm.com> (serveur IBM aux Etats-Unis)

*Compagnie IBM France  
Direction Qualité  
17, avenue de l'Europe  
92275 Bois-Colombes Cedex*

Cette édition s'applique à la version 9.1.0 de IBM Campaign et à toutes les éditions et modifications ultérieures jusqu'à mention contraire dans les nouvelles éditions.

© Copyright IBM Corporation 1998, 2013.

# Table des matières

## Avis aux lecteurs canadiens . . . . . v

## Chapitre 1. Introduction . . . . . 1

Récapitulatif d'API CampaignServices . . . . .	1
Implémentations à l'aide de l'API client . . . . .	1
Fichier .jar de l'API client . . . . .	2
Fichiers .jar dépendants. . . . .	2
Constructeur de l'API client . . . . .	2
Constructeurs paramétrés des classes de prise en charge . . . . .	2
Implémentations directement à l'aide du langage WSDL . . . . .	2
Emplacement WSDL et URL de service . . . . .	3
Génération de modules de remplacement et de classes . . . . .	3
Utilisation de modules de remplacement et de classes de prise en charge générés . . . . .	3
Références . . . . .	4
Exigences pour l'API CampaignServices . . . . .	4
Présentation de la conception. . . . .	5
Informations sur l'API CampaignServices. . . . .	5

## Chapitre 2. Types de données . . . . . 9

WSReference . . . . .	9
WSVersion . . . . .	9
WSServiceInfo . . . . .	9
WSAttributeTypeEnum . . . . .	10
WSAttributeStatusEnum . . . . .	10
WSAccessTypeEnum . . . . .	10
WSSelectTypeEnum . . . . .	10
WSRunStatusEnum . . . . .	10
WSRunTypeEnum . . . . .	11
WSAttribute . . . . .	11
WSAttributeMetadata . . . . .	12
WSCampaignInfo . . . . .	14
WSComponentOrFolderInfo . . . . .	14
WSTargetCellInfo . . . . .	14
WSMetricsInfo . . . . .	14
WSRunResults . . . . .	15
WSOfferInfo . . . . .	15
WSOfferCodeOrName . . . . .	15
WSOfferValidationInfo . . . . .	16
WSOfferTemplateInfo . . . . .	16
WSBulkOfferInfo . . . . .	16
WSOfferInfoStatus . . . . .	16

## Chapitre 3. Exceptions courantes . . . . . 17

RemoteException . . . . .	17
AuthenticationException . . . . .	17
AuthorizationException . . . . .	17
DataException . . . . .	17
LockException . . . . .	17
InvalidComponentException. . . . .	18
InvalidAttributeException . . . . .	18
AttributeNotFoundException . . . . .	18

AttributeExistsException . . . . .	18
CompositeException . . . . .	18

## Chapitre 4. Méthodes d'API CampaignServices. . . . . 19

Méthodes de service . . . . .	19
getServiceInfo . . . . .	19
Attributs . . . . .	19
getAttributesByName . . . . .	19
updateAttributes . . . . .	20
getAttributeMetadataByName . . . . .	21
createAttributeMetadata . . . . .	22
updateAttributeMetadata . . . . .	23
deleteAttributeMetadata . . . . .	24
Campagnes . . . . .	25
generateCampaignCode . . . . .	26
deleteCampaigns . . . . .	27
createCampaign . . . . .	27
listCampaignsByPage . . . . .	28
Méthodes de population ciblée . . . . .	29
createTargetCell . . . . .	30
bulkCreateTargetCells . . . . .	31
listTargetCells . . . . .	32
bulkUpdateTargetCells. . . . .	33
getRunResultsByCell . . . . .	34
bulkDeleteTargetCells . . . . .	34
updateTemplateAttributes . . . . .	35
listBottomUpTargetCells . . . . .	36
Analyse . . . . .	36
getCampaignMetrics . . . . .	36
Offre, méthodes de liste d'offres et de modèle d'offre 37	
listOffersAndFolders . . . . .	37
searchOffersBasic . . . . .	38
listOffersByPage . . . . .	39
createSmartOfferList . . . . .	40
createStaticOfferList . . . . .	41
getOffers . . . . .	42
validateOffers . . . . .	42
editOfferList . . . . .	43
createOffer. . . . .	44
retireOffers . . . . .	45
deleteOffers . . . . .	45
deleteOffersAndLists . . . . .	46
listOfferTemplates . . . . .	46
createTemplate . . . . .	47
getOfferTemplate . . . . .	47
retireOfferTemplates . . . . .	48
getOffersAndListsByPage. . . . .	48
bulkCreateOffers . . . . .	49
getOfferListDetails . . . . .	49
getOfferListMembers . . . . .	50
getOffersByQuery . . . . .	50
retireOfferLists . . . . .	51
createFolder . . . . .	52
editFolder . . . . .	52

getSubFoldersList . . . . .	53
moveFolders . . . . .	54
deleteFolders . . . . .	54
<b>Chapitre 5. Utilisation de l'API. . . . .</b>	<b>57</b>
Utilisation du fichier .jar de l'API client . . . . .	57
OfferAPI.java . . . . .	57
Utilisation du fichier WSDL . . . . .	59
Considérations relatives aux performances . . . . .	59

Packaging . . . . .	60
<b>Avant de contacter le support technique IBM . . . . .</b>	<b>61</b>
<b>Remarques . . . . .</b>	<b>63</b>
Marques . . . . .	65
Règles de confidentialité et conditions d'utilisation . . . . .	65

---

## Avis aux lecteurs canadiens

Le présent document a été traduit en France. Voici les principales différences et particularités dont vous devez tenir compte.

### Illustrations

Les illustrations sont fournies à titre d'exemple. Certaines peuvent contenir des données propres à la France.

### Terminologie

La terminologie des titres IBM peut différer d'un pays à l'autre. Reportez-vous au tableau ci-dessous, au besoin.

IBM France	IBM Canada
ingénieur commercial	représentant
agence commerciale	succursale
ingénieur technico-commercial	informaticien
inspecteur	technicien du matériel

### Claviers

Les lettres sont disposées différemment : le clavier français est de type AZERTY, et le clavier français-canadien de type QWERTY.








### OS/2 et Windows - Paramètres canadiens

Au Canada, on utilise :

- les pages de codes 850 (multilingue) et 863 (français-canadien),
- le code pays 002,
- le code clavier CF.

### Nomenclature

Les touches présentées dans le tableau d'équivalence suivant sont libellées différemment selon qu'il s'agit du clavier de la France, du clavier du Canada ou du clavier des États-Unis. Reportez-vous à ce tableau pour faire correspondre les touches françaises figurant dans le présent document aux touches de votre clavier.

<b>France</b>	<b>Canada</b>	<b>Etats-Unis</b>
 (Pos1)		Home
Fin	Fin	End
 (PgAr)		PgUp
 (PgAv)		PgDn
Inser	Inser	Ins
Suppr	Suppr	Del
Echap	Echap	Esc
Attn	Intrp	Break
Impr écran	ImpEc	PrtSc
Verr num	Num	Num Lock
Arrêt défil	Défil	Scroll Lock
 (Verr maj)	FixMaj	Caps Lock
AltGr	AltCar	Alt (à droite)

### **Brevets**

Il est possible qu'IBM détienne des brevets ou qu'elle ait déposé des demandes de brevets portant sur certains sujets abordés dans ce document. Le fait qu'IBM vous fournisse le présent document ne signifie pas qu'elle vous accorde un permis d'utilisation de ces brevets. Vous pouvez envoyer, par écrit, vos demandes de renseignements relatives aux permis d'utilisation au directeur général des relations commerciales d'IBM, 3600 Steeles Avenue East, Markham, Ontario, L3R 9Z7.

### **Assistance téléphonique**

Si vous avez besoin d'assistance ou si vous voulez commander du matériel, des logiciels et des publications IBM, contactez IBM direct au 1 800 465-1234.

---

## Chapitre 1. Introduction

**Remarque :** Ce document définit la version 3.0 de l'interface de programme d'application des services IBM® Campaign, également appelée CampaignServices.

**Remarque :** Si vous effectuez la mise à niveau vers IBM Campaign version 8.6 ou ultérieure et si l'API des services IBM Campaign est actuellement implémentée, les changements de l'API dus à la mise à niveau d'AXIS version 1.4.1 vers AXIS2 1.5.2 nécessitent d'apporter des changements à votre code d'application. Pour plus de détails, voir «Fichiers .jar dépendants», à la page 2.

Cette section comprend les rubriques suivantes :

- «Récapitulatif d'API CampaignServices»
- «Implémentations à l'aide de l'API client»
- «Implémentations directement à l'aide du langage WSDL», à la page 2
- «Références», à la page 4
- «Exigences pour l'API CampaignServices», à la page 4
- «Présentation de la conception», à la page 5

---

### Récapitulatif d'API CampaignServices

- Le moteur SOAP est AXIS2 1.5.2.
- Le langage WSDL est restructuré pour traiter les problèmes de gestion des paramètres obligatoires/facultatifs.
- Le fichier .jar de l'API client a été modifié suite aux changements de langage WSDL. Ainsi, les modules de remplacement et les classes générés sont modifiés. Les paramètres de la méthode d'API client n'ont pas été modifiés, mais les constructeurs des objets de valeur de prise en charge sont modifiés en raison de l'utilisation du convertisseur AXIS2 WSDL2Java.
- L'URL de service Web pointe vers :

`http://<host>:<port>/Campaign/services/CampaignServices30Service`

et le fichier WSDL correspondant est récupéré à l'adresse :

`http://<host>:<port>/Campaign/services/CampaignServices30Service?wsdl`

Si vous utilisez actuellement l'API, vous devez modifier votre code d'application. Selon que vous utilisez l'API client ou le langage WSDL, reportez-vous aux sections suivantes pour plus de détails :

- «Implémentations à l'aide de l'API client»
- «Implémentations directement à l'aide du langage WSDL», à la page 2

---

### Implémentations à l'aide de l'API client

Si vous utilisez le fichier .jar de l'API client pour interagir avec l'application Web Campaign, les sections suivantes détaillent les informations importantes.

## Fichier .jar de l'API client

L'application Java™ doit utiliser le fichier .jar qui se trouve dans :

```
<CAMPAIGN_HOME>/devkits/CampaignServicesAPI/lib/  
CampaignServicesClient30.jar
```

Pour obtenir un exemple Java illustrant la création d'une offre, voir «OfferAPI.java», à la page 57. Ce même exemple est disponible dans votre installation de Campaign sous :

```
<CAMPAIGN_HOME>/devkits/CampaignServicesAPI/samples/OfferAPI.java
```

## Fichiers .jar dépendants

Suite à la mise à niveau vers AXIS2 version 1.5.2, votre application Java doit également effectuer la mise à niveau qui consiste à utiliser les fichiers .jar de distribution AXIS2 1.5.2, car CampaignServicesClient30.jar est dépendant de ces fichiers .jar. Tous les fichiers .jar dépendants doivent être inclus dans le chemin de classes Java de votre application et sont disponibles dans le fichier Campaign.war qui se trouve dans <CAMPAIGN\_HOME>/Campaign.war.

Vous devez extraire les fichiers .jar de Campaign.war, puis les inclure dans le chemin d'accès aux classes Java.

## Constructeur de l'API client

Lors de la construction de l'objet API client, changez l'URL du service Web et la signature d'exception, comme indiqué dans l'exemple suivant.

```
try {  
    URL serviceURL = new URL(PROTOCOL, HOST, PORT,  
        "/Campaign/services/CampaignServices30Service");  
    CampaignServices30SoapClient client = new  
    CampaignServices30SoapClient(serviceURL, TIMEOUT);  
} catch (RemoteException exception) {  
    exception.printStackTrace();  
}
```

## Constructeurs paramétrés des classes de prise en charge

Avec le moteur AXIS2, les classes et les modules de remplacement générés ne disposent pas de constructeurs paramétrés. Au lieu de cela, ces classes disposent uniquement du constructeur sans argument par défaut, avec des méthodes d'accès set et get pour les membres.

```
WSReference wsRef = new WSReference();  
wsRef.setComponentTypeEnum(typeEnum);  
wsRef.setId(id);
```

---

## Implémentations directement à l'aide du langage WSDL

Le langage WSDL du service Web Campaign est utilisé pour générer des modules de remplacement côté client et des classes de prise en charge à l'aide de n'importe quel outil de conversion tiers. Si vous utilisez le langage WSDL pour interagir avec l'application Web Campaign, les sections suivantes détaillent les informations importantes. Les exemples fournis ici utilisent l'outil WSDL2Java d'Apache AXIS2 1.5.2.



## Emplacement WSDL et URL de service

Le service Web Campaign pour IBM Campaign est déployé à l'adresse suivante :

`http://host:port/Campaign/services/CampaignServices30Service`

Le fichier WSDL correspondant peut être récupéré à l'adresse suivante :

`http://host:port/Campaign/services/CampaignServices30Service?wsdl`

## Génération de modules de remplacement et de classes

L'outil WSDL2Java d'Apache AXIS2 1.5.2 peut être utilisé pour générer les modules de remplacement et les classes Java de prise en charge à partir du langage WSDL. Un exemple de tâche Ant est présenté ici.

L'outil peut également être utilisé à partir de la ligne de commande avec ce même ensemble d'arguments. Les valeurs d'argument peuvent être modifiées en fonction de votre environnement.

**Remarque :** La liaison ADB par défaut est utilisée pour l'exemple de convertisseur WSDL2Java suivant.

```
<java classname="org.apache.axis2.wsdl.WSDL2Java" fork="true">
  <classpath refid="axis2.class.path"/> <!--Class path having
AXIS2 libraries -->
  <arg value="-uri"/>
  <arg file="CampaignServices30.wsdl"/> <!--Actual location of
WSDL -->
  <arg value="-s"/> <!-- Generate sync style code -->
  <arg value="-Euwc"/> <!-- Takes care of generating Wrapper
java types for nillable = true elements. -->
  <arg value="-uw"/> <!-- Unwrap params -->
  <arg value="-u"/> <!-- Unpack classes -->
  <arg value="-ns2p"/> <!-- Namespace to package mapping. Customer
can have their own package names. -->
  <arg value="http://webservices.unica.com/campaign/CampaignServices/
3.0=com.unica.publicapi.campaign.campaignservices.soap.v30"/>
  <arg value="-o"/> <!-- Output directory -->
  <arg file="${autogen.java.dir}"/>
</java>
```

## Utilisation de modules de remplacement et de classes de prise en charge générés

Le module de remplacement peut être utilisé comme suit :

```
CampaignServices30ServiceStub serviceStub = new
CampaignServices30ServiceStub(serviceURL);

serviceStub._getServiceClient().getOptions().setTimeOutInMilliseconds
(webServiceTimeout); //Timeout in milliseconds.
```

L'offre peut être créée comme suit :

```
try{
  //Please change host and port to match your environment.
  String serviceURL = "http://host:port/Campaign/services/CampaignServices30Service";
  CampaignServices30ServiceStub serviceStub = new CampaignServices30ServiceStub(serviceURL);
  long webServiceTimeout = 2*60*1000; // 2 minutes
  serviceStub._getServiceClient().getOptions().setTimeOutInMilliseconds(webServiceTimeout); //Timeout in milliseconds.

  WSTextAttribute nameAttribute = new WSTextAttribute();
  nameAttribute.setMetadata(null);
  nameAttribute.setName("uacOfferDescription");
  nameAttribute.setValues(new String[]{"description " + System.currentTimeMillis()});
```

```

WSTextAttribute[] wsAttributes = {nameAttribute};
// convert to WSAttributeArrays
WSAttributeArrays obj = new WSAttributeArrays();
obj.setTextAttributes(wsAttributes);
//Please change the values of following variables to match your environment.
String authorizationLoginName = "asm_admin"; //login user name
String partitionName = "partition1"; //Use your security policy of Campaign
String securityPolicyName = "Global Policy"; //Use your security policy of Campaign

String offerName = "1st Offer"; //Name of the offer to be created.
String templateName = "Offer Template"; //Existing offer template name.
long folderID = 100; //Actual ID of the folder where this offer will be created.
//For folderID <=0, offer will be created at root level.

CreateOffer createOfferObject = new CreateOffer();
createOfferObject.setAuthorizationLoginName(authorizationLoginName);
createOfferObject.setPartitionName(partitionName);
createOfferObject.setRequestedLocale(Locale.US.toString());
createOfferObject.setSecurityPolicyName(securityPolicyName);
createOfferObject.setName(offerName);
createOfferObject.setFolderID(folderID);
createOfferObject.setTemplateName(templateName);
createOfferObject.setAttributes(obj);
// make campaign Webservice call
WSCreateOfferResponse wsResponse = serviceStub.createOffer(createOfferObject);
// process status
WSRequestStatus status = wsResponse.getStatus();
// done
WSOfferInfo offerInfo = wsResponse.getOfferInfo();
System.out.println("status = "+status.getStatusType());
System.out.println("offerInfo = "+offerInfo.getName());
} catch (Exception exception) {
//Handle the Exception here.
exception.printStackTrace();
}
}

```

Dans cet exemple, `createOffer()` accepte désormais un seul paramètre de type `CreateOffer`.

Avec le moteur AXIS2, les classes et les modules de remplacement générés ne disposent plus de constructeurs paramétrés. Pour plus de détails, voir «Constructeurs paramétrés des classes de prise en charge», à la page 2.

---

## Références

Les références suivantes ont été utilisées pour préparer cette spécification :

- "Basic Profile Version 1.1", Organisation WS-I (Web Service Interoperability), 10 avril 2006. (<http://www.ws-i.org/Profiles/BasicProfile-1.1-2006-0310.html>)
- "SOAP 1.2 (draft)", groupe de travail W3C Soap, 24 juin 2003 (<http://www.w3.org/TR/soap/>)
- "JAX-RPC 1.1", Sun Microsystems, 14 octobre 2003 (<http://java.sun.com/webservices/jaxrpc/index.jsp>)
- Groupe de travail des services Web Apache (<http://ws.apache.org/axis2>)

---

## Exigences pour l'API CampaignServices

De par sa conception, l'API CampaignServices doit exécuter les opérations suivantes :

- fournir des droits précis de création, reconnaissance, lecture et mise à jour des composants IBM Campaign tout en isolant les clients des détails de mise en oeuvre sous-jacents ;
- travailler en parallèle avec l'interface utilisateur IBM Campaign existante avec des effets réduits au minimum
- garantir la validité des données

- assurer les services de sécurité requis d'IBM Campaign
- prendre en charge le protocole SOAP normalisé, y compris l'authentification sécurisée.

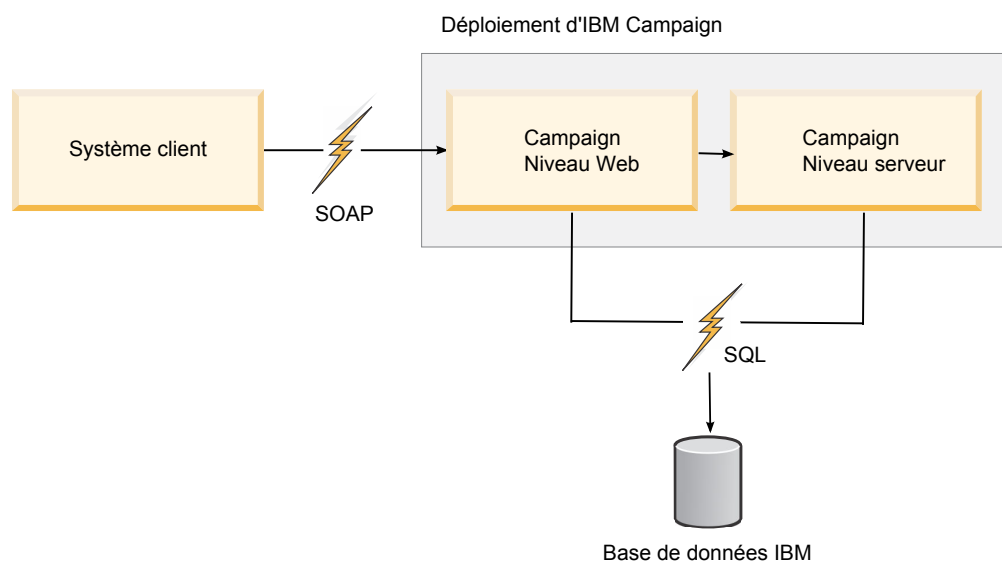
## Présentation de la conception

L'API CampaignServices est une façade qui offre une vue client d'une instance d'application IBM Campaign en cours d'exécution. Seul un sous-ensemble des fonctions d'IBM Campaign est présenté, mais il suffit à piloter les aspects clés des fonctionnalités Campaign. L'API est utilisée simultanément avec des utilisateurs Web IBM Campaign et d'autres threads de l'API.

En règle générale, l'API prend en charge les types d'opérations suivants sur les campagnes, les offres et les composants de population ciblée :

- Création de composant
- Reconnaissance de composant
- Suppression de composant
- Création, inspection et changement des métadonnées d'attribut et attributs de composant
- Extraction des résultats de l'exécution du diagramme

Le diagramme suivant présente un exemple de déploiement de CampaignServices 3.0.



## Informations sur l'API CampaignServices

Cette section traite de points particuliers et d'informations générales sur la conception de l'API CampaignServices.

### Versions et compatibilité amont

Les futures versions de l'API CampaignServices sont compatibles avec les versions antérieures et toutes les éditions secondaires et de maintenance qui partagent le même numéro de version. Toutefois, IBM se réserve le droit d'abandonner la compatibilité avec la version antérieure pour les éditions principales "point zéro" (x.0).

Le numéro de version de cette API est incrémenté si l'un des changements suivants est apporté :

- changement de l'interprétation des données ;
- changement de la logique applicative, c'est-à-dire la fonctionnalité de la méthode service ;
- changement des paramètres de méthode et/ou des types de retour.

Le numéro d'édition de l'API est incrémenté si l'un des changements suivants est apporté.

- ajout d'une nouvelle méthode ;
- nouveau type de données ajouté et son utilisation limitée aux nouvelles méthodes ;
- ajout d'un nouveau type à un type énuméré ;
- définition d'une nouvelle version d'interface.

IBM continue à prendre en charge le langage WSDL publié, le client SOAP et la version d'Apache Axis utilisée pour implémenter l'offre SOAP au moins jusqu'à la prochaine version d'IBM. En pratique, cela correspond à la prise en charge simultanée de plusieurs services Web propres à la version. (IBM prend déjà en charge plusieurs versions de ce service en interne.)

## Authentification d'utilisateur

L'authentification consiste à établir l'identité d'un utilisateur.

**Remarque :** Pour cette édition, l'authentification d'utilisateur est de la responsabilité de l'application client.

## Autorisation utilisateur

L'autorisation concerne les droits dont un utilisateur authentifié dispose concernant les composants et les opérations exposés par l'API.

Il est possible qu'un utilisateur soit correctement authentifié mais ne dispose pas de droits suffisants pour effectuer certaines opérations, telles que la changement des informations récapitulatives d'une campagne. Dans ce cas, la méthode d'API émet `AuthorizationException`.

## Locale

Les requêtes API fournissent un paramètre **requestedLocale** facultatif qui définit les paramètres régionaux à utiliser pour effectuer cette requête particulière. S'il n'est pas défini, l'API utilise par défaut les paramètres régionaux préférés de l'utilisateur IBM. L'algorithme de mise en correspondance optimale Java habituel est utilisé pour renvoyer des messages et tout autre texte localisé selon les paramètres régionaux demandés.

Ce paramètre fait partie de la classe de type `java.util.Locale`.

**Remarque :** Certains textes spécifiés par l'utilisateur, tels que les descriptions de campagne, sont dans les paramètres régionaux de l'utilisateur qui a spécifié le texte. IBM Campaign ne tente pas de localiser ces données. Seuls les informations, les avertissements et les messages d'erreur sont localisés par l'API.

## Gestion d'état

L'API CampaignServices est sans état, ce qui signifie qu'aucune information par client n'est enregistrée par l'API au travers des appels.

Bien entendu, des appels API spécifiques peuvent changer l'état des instances de composant sous-jacentes gérées par Campaign, et ces changements d'état peuvent être conservés dans la base de données.



---

## Chapitre 2. Types de données

Cette section définit les types de données publiques utilisés par l'API CampaignServices.

---

### WSReference

Encapsuleur simple autour d'un identificateur de base de données :

- **componentTypeEnum** : type énuméré qui indique le type de composant correspondant à l'ID, à savoir :
  - FOLDER
  - CAMPAIGN
  - FLOWCHART
  - TCS\_CELL
  - OFFER
  - OFFER\_LIST
  - OFFER\_TEMPLATE
- *id* : *long* définissant un identificateur numérique, unique et propre à la base de données pour la référence.

---

### WSVersion

Type d'encapsuleur capturant les divers composants d'une version, notamment :

- *major* : entier définissant le numéro de version, par exemple '8' pour la version complète 8.1.2.3.
- *minor* : entier définissant le numéro d'édition, par exemple '1' pour la version complète 8.1.2.3.
- *maintenance* : entier facultatif définissant le numéro de maintenance de la version (le cas échéant), par exemple '2' pour la version complète 8.1.2.3. Jamais fourni avec une version de l'API.
- *patch* : entier facultatif définissant le numéro du correctif (le cas échéant), par exemple '3' pour la version complète 8.1.2.3. Jamais fourni avec une version de l'API.

---

### WSServiceInfo

Type d'encapsuleur simple autour des informations relatives au service. Contient les zones suivantes :

- *apiVersion* : instance *WSVersion* définissant la version la plus récente de l'API prise en charge par le service. (*apiVersion* inclut uniquement les informations relatives aux versions et éditions.)
- *campaignVersion* : instance *WSVersion* définissant la version complète de l'instance d'IBM Campaign sous-jacente.
- *name* : nom interne du service, tel que "CampaignServices30Service".

---

## WSAttributeTypeEnum

Type énuméré définissant tous les types d'attribut possibles, à savoir :

- STANDARD : attribut standard ou de base défini par Campaign.
- CUSTOM : attribut défini par une autre application IBM , le client ou un tiers.
- INPUT\_PARAMETER : paramètre d'entrée, tel qu'un attribut utilisé pour exécuter un diagramme IBM Campaign.
- OUTPUT\_PARAMETER : paramètre de sortie, tel qu'un attribut dont la valeur est complétée suite à l'exécution d'un diagramme dans IBM Campaign.

---

## WSAttributeStatusEnum

Enumération de tous les codes de statut d'attribut possibles, à savoir :

- ACTIVE : l'attribut est actif et peut être utilisé à volonté.
- RETIRED : l'attribut est supprimé du service et ne doit pas être utilisé.

---

## WSAccessTypeEnum

Type énuméré définissant tous les types d'accès possibles à la valeur d'attribut, à savoir :

- READ\_ONLY : la valeur d'attribut peut être lue et affichée, mais pas modifiée.
- READ\_WRITE : la valeur d'attribut peut être lue, affichée et modifiée.

L'accès aux attributs s'ajoute aux droits de sécurité. Par exemple, si la stratégie de sécurité pour l'utilisateur client refuse l'accès en lecture à un attribut particulier, l'accès aux attributs ne peut pas remplacer ce paramètre de sécurité. En fait, dans ce cas, l'API ne renvoie jamais l'attribut au client.

---

## WSSelectTypeEnum

Définit tous les types de sélection possibles pour une valeur d'attribut particulière, à savoir :

- NONE : aucune sélection (*hasOptions* a la valeur false).
- SINGLE\_SELECT : une seule option d'attribut peut être sélectionnée à la fois dans la liste des options possibles (valide uniquement pour un attribut *hasOptions*).
- MULTIPLE\_SELECT : semblable à SINGLE\_SELECT, sauf qu'une ou plusieurs options peuvent être sélectionnées à la fois.

---

## WSRunStatusEnum

Type énuméré de tous les états d'exécution de diagramme, branche ou cible possibles, à savoir :

- NOT\_STARTED : l'exécution est planifiée, mais n'a pas encore démarré.
- RUNNING : exécution en cours.
- CANCELLED : l'exécution a été annulée, soit par un utilisateur de Campaign soit par cette API.
- SUCCEEDED : l'exécution a abouti.
- FAILED : l'exécution a échoué ; les erreurs sont détaillées séparément. (Voir «WSRunResults», à la page 15.)



---

## WSRunTypeEnum

Type énuméré de tous les types d'exécution possibles, à savoir :

- NOT\_RUN
- TEST\_RUN
- PRODUCTION\_RUN
- RUN\_SKIPPED
- TEST\_FLOWCHART
- PRODUCTION\_FLOWCHART
- TEST\_BRANCH
- PRODUCTION\_BRANCH
- TEST\_PROCESS
- PRODUCTION\_PROCESS

---

## WSAttribute

Les attributs offrent un mécanisme simple, extensible permettant de lier des données arbitraires à des instances de composant accessibles via l'API, qu'il s'agisse de données standard comme le *nom* d'une campagne, de paramètres d'entrée d'exécution de diagramme comme le *genre* ou de données personnalisées arbitraires spécifiées par une autre application IBM ou un client IBM.

**Remarque :** Dans cette API, les attributs sont utilisés pour modéliser la plupart des données de composant, pas seulement les attributs personnalisés de Campaign.

De nombreux attributs sont généralement associés aux composants et exposés par l'API CampaignServices sous la forme d'une mappe spécialement typée appelée *AttributeMap*. Les données d'attribut sont représentées sous la forme d'une classe concrète typée dans toute l'API, par exemple *WSDecimalAttribute*, pour les attributs qui contiennent des données décimales (numériques double précision).

Chaque attribut inclut les éléments suivants :

- *Nom* : nom unique de l'attribut. Ce nom sert de clé pour accéder à l'attribut et à ses métadonnées dans l'instance de composant où il apparaît. Le format du nom est non défini ; dans certains cas, il est affecté par le service, par le client, ou par un utilisateur IBM Campaign.

En principe, ce nom ne correspond pas au nom d'affichage présenté à un utilisateur client ou Campaign. Il peut être normalisé par l'API, par exemple *uacDescription*, affecté par IBM Campaign lors de la publication de diagrammes, ou affecté par le client ou l'application IBM lors de la définition d'attributs personnalisés. Dans tous les cas, cependant, le nom est garanti unique.

- *Métadonnées* : informations (facultatives) sur les données de l'attribut, telles que le type de données de la valeur, le nom d'affichage, la description, les invites, la valeur par défaut, le type de sélection, la longueur (texte), la précision (décimales), les options (en cas de sélection unique ou multiple), etc. Voir «WSAttributeMetadata», à la page 12.
- *Valeurs* : matrice de zéro ou plusieurs objets de valeur typés. La zone de valeurs est fournie par la classe d'attributs concrète ; le type de chaque valeur doit être identique et en accord avec la définition de type de la zone de métadonnées de l'attribut. Néanmoins, les attributs ne prennent pas tous en charge plusieurs valeurs.

Les types d'attributs concrets suivants sont pris en charge :

- **WSBooleanAttribute** : attribut dont la valeur est booléenne, c'est-à-dire *true* ou *false*.
- **WSIntegerAttribute** : valeur entière (*java.lang.Long*).
- **WSDecimalAttribute** : valeur numérique décimale à double précision (*java.lang.Double*).
- **WSCurrencyAttribute** : valeur de devise composée, incluant un code de devise ISO 4217 facultatif de la valeur de devise, tel que "USD" pour le dollar américain, et les valeurs de devise capturées comme *Double*. Si le code de devise n'est pas indiqué, la valeur par défaut utilisée par IBM Campaign est appliquée.  
Voir <http://www.xe.com/symbols.php> pour obtenir une liste des pays, des symboles monétaires et des codes. Les paramètres régionaux utilisés pour une valeur de devise peuvent être différents des paramètres régionaux préférés de l'utilisateur.
- **WSCalendarAttribute** : dont les valeurs sont des dates de calendrier, ou des dates/heures, d'un fuseau horaire et de paramètres régionaux donnés.
- **WSTextAttribute** : chaîne de texte Unicode (éventuellement nulle ou vide).

**Remarque :** La liste des attributs possibles est généralement différente pour chaque type de composant, mais les listes peuvent se chevaucher.

---

## WSAttributeMetadata

WSAttributeMetadata définit les informations relatives aux données d'un attribut de type particulier, tel que le type de données de la valeur, un texte localisé (nom d'affichage, description, invites), sa valeur par défaut, la plage de valeurs autorisées, le type de sélection, les options (en cas de sélection unique ou multiple). Comme avec les attributs, les métadonnées d'attribut sont typées. Par exemple, un WS *DecimalAttributeMetadata* doit être lié à un *WSDecimalAttribute myNumber*, et toutes les valeurs, y compris les valeurs d'attribut, la valeur par défaut des métadonnées et les valeurs d'option éventuelles, sont de type *Double*.

Les descriptions, les étiquettes et autres textes des métadonnées d'attribut sont localisés ; par contre, il se peut que le texte spécifié par l'utilisateur soit disponible uniquement tel qu'il a été saisi par l'utilisateur. Chaque appel API inclut des paramètres régionaux demandés que le code client peut utiliser pour définir les paramètres régionaux qu'un utilisateur donné souhaite utiliser pour l'affichage des messages localisés. Les règles de rétro migration habituelles des paramètres régionaux Java sont utilisées pour satisfaire la demande.

WSAttributeMetadata inclut les zones suivantes :

- *name* : nom de l'attribut, standard ou personnalisé ; correspond également au nom utilisé par l'attribut de liaison à ces métadonnées. Les attributs standard sont définis par le système et ont des noms standard dans un espace de nom réservé (c'est-à-dire qu'ils utilisent un préfixe "uac") ; les noms personnalisés peuvent utiliser une autre convention de dénomination.

**Remarque :** Le nom d'attribut doit être unique, n'est jamais localisé et fait l'objet de restrictions en termes de longueur (selon la base de données et le contenu des caractères). Le nom n'est pas sensible à la casse et peut être composé de n'importe quelle combinaison de caractères Unicode (lettres ou chiffres) ainsi que du caractère de soulignement '\_', mais il ne peut pas commencer par un chiffre.

- *description* : description facultative de l'attribut. Adaptée pour une infobulle ou une autre présentation de l'interface utilisateur.
- Prédicats : ensemble de prédicats décrivant l'attribut :
  - *isRequired* : true si l'attribut est obligatoire.
  - *isInternal* : true si l'attribut est défini par le système et destiné à un usage interne uniquement (ne doit pas être présenté à un utilisateur).
  - *isGenerated* : true si les valeurs de l'attribut sont générées automatiquement par IBM Campaign lorsque le composant est créé, par exemple un code de population ciblée. En général, le *accessTypeEnum* sera READ\_ONLY pour les valeurs générées.
  - *hasOptions* : true si l'attribut comporte des options. Implique que des options sont définies pour cette métadonnée et que le *selectTypeEnum* est SINGLE\_SELECT ou MULTIPLE\_SELECT.
- *typeEnum* : *WSAttributeTypeEnum* définissant le type d'attribut, comme STANDARD ou CUSTOM.
- *statusEnum* : *WSAttributeStatusEnum* définissant le statut de l'attribut, par exemple ACTIVE.
- *accessTypeEnum* : *WSAccessTypeEnum* définissant le type d'accès à la valeur d'attribut, par exemple READ\_ONLY.
- *selectTypeEnum* : *WSAccessTypeEnum* qui définit le type de sélection qui est utilisé pour l'attribut, comme SINGLE. Doit être NONE pour les attributs de campagne et de cible ou si aucune option n'est fournie.
- *componentTypeEnum* : *WSComponentTypeEnum* de tous les composants Campaign possibles exposés par l'API, par exemple CAMPAIGN, FOLDER.
- *defaultValue* (diagrammes uniquement) : valeur par défaut du type facultative pour l'attribut. Cette valeur est fournie par la classe de métadonnées d'attribut concrète ; par exemple la valeur par défaut de *WSTextAttributeMetadata* est de type chaîne. (Reportez-vous à la description des valeurs d'attribut). Pour les composants autres que les diagrammes, la valeur par défaut est non définie.
- *options* : liste facultative des options pour cet attribut. Ensemble, les options d'un attribut définissent l'ensemble exact des valeurs autorisées pour cet attribut ; chaque option est typée. Ainsi, par exemple, seul *WSTextAttributeOption* peut être lié à *WSTextAttributeMetadata*.

**Remarque :** Une restriction s'applique aux options ; seuls les attributs de texte sont pris en charge.

Chaque option définit les éléments suivants :

- *prompt* : invite de l'option appropriée pour les menus déroulants, par exemple "Male" comme option d'attribut de genre. Contrairement à l'invite de métadonnées, les noms d'affichage d'option n'incluent généralement aucune ponctuation.
- *description* : description localisée de l'option, par exemple "Une personne de sexe masculin". Adaptée au texte d'infobulle.
- *isDefault* : true si cette option particulière est la valeur par défaut. Pour les types de sélection MULTIPLE\_SELECT, plusieurs options peuvent être marquées comme valeur par défaut.
- *value* : valeur d'option typée. Comme pour la métadonnée d'attribut *defaultValue*, cette valeur est fournie par la sous-classe d'option concrète ; par exemple, la valeur de *WSDecimalAttributeOption* est de type décimal. (Reportez-vous à la description des valeurs d'attribut). Si l'on reprend l'exemple

du *genre* ci-dessus, la valeur pourrait être déclarée comme une chaîne "m" (*WSTextAttributeOption*) ou comme un code numérique 123 (*WSDecimalAttributeOption*).

---

## WSCampaignInfo

Type d'encapsuleur simple autour des données d'attribut de campagne.

Contient les zones suivantes :

- *reference* : référence de la campagne.
- *name* : nom de campagne (*uacName*) ; non garanti unique.
- *description* : description de campagne facultative (*uacDescription*).
- *campaignCode* : code de campagne unique (*uacCampaignCode*) ; affecté par le client ou Campaign.

---

## WSComponentOrFolderInfo

Contient une combinaison de données d'attribut de campagne ou de dossier encapsulées, telles que le nom d'affichage, sa référence.

Contient les zones suivantes :

- *reference* : référence du composant ou du dossier.
- *name* : nom de composant ou de dossier (*uacName*) ; non garanti unique.
- *description* : description de composant ou de dossier facultative (*uacDescription*).
- *componentCode* : code unique pour le composant, ou NULL pour un dossier.

---

## WSTargetCellInfo

Encapsuleur simple autour des données d'attribut de ligne de population ciblée.

Contient les zones suivantes :

- *reference* : référence de la cible.
- *name* : nom de la cible (*uacName*) ; non garanti unique.
- *description* : description de cible facultative (*uacDescription*).
- *cellCode* : code de cible (*uacCellCode*) ; affecté par le client ou Campaign. Vous pouvez imposer l'unicité des codes de cible en définissant la valeur false pour le paramètre de configuration *DuplicateCellCodesAllowed* d'IBM Campaign.
- *flowchartName* : nom facultatif du diagramme auquel la cible est liée.

---

## WSMetricsInfo

Type d'encapsuleur simple autour des données analytiques de campagne, nombre de contacts inclus. Contient les zones suivantes :

- *totalContacts* : long indiquant le nombre total de contacts.
- *responses* : liste typée d'instances *WSMetricsResponse*, chaque instance qui définit des informations de contact pour une réponse :
  - *typeCode* : chaîne qui définit le code de type de réponse, tel que *PHC* pour un contact téléphonique.
  - *count* : long indiquant le nombre de fois où ce contact est apparu.

---

## WSRunResults

Type d'encapsuleur autour des résultats de l'exécution d'un diagramme, d'une zone de processus ou d'une cible (éventuellement encore en cours) comprenant le statut d'exécution, les dates/heures de début et de fin de l'exécution du diagramme, et le nombre.

Contient les zones suivantes :

- *sourceReference* : référence facultative de la source du résultat de l'exécution. Selon le contexte de l'extraction des résultats d'exécution, elle peut porter sur un diagramme, une zone de processus de diagramme ou une population ciblée. Dans tous les cas, les données de résultat d'exécution restantes font référence à cette source.
- *flowchartName* : nom du diagramme qui a été exécuté.
- *flowchartId* : identificateur de base de données pour le diagramme.
- *runId* : identificateur de base de données de l'exécution.
- *typeEnum* : type énuméré définissant l'exécution qui a généré les résultats, par exemple PRODUCTION\_PROCESS (voir *WSRunTypeEnum*).
- *statusEnum* : type énuméré qui définit le statut d'exécution, par exemple RUNNING (voir *WSRunStatusEnum*).
- *statusCode* : code de statut entier facultatif.
- *statusMessage* : message de statut facultatif.
- *startDate* : date-heure facultative du démarrage de l'exécution ; elle est null si l'exécution n'a pas démarré.
- *endDate* : de la même manière que *startDate*, date-heure de fin de l'exécution (réussite ou échec) ; elle est nulle si l'exécution n'a pas démarré ou n'est pas encore terminée.
- *count* : nombre total facultatif de contacts sélectionnés par l'exécution ; peut être égal à zéro ou NULL si l'exécution n'est pas terminée.

---

## WSOfferInfo

Type d'encapsuleur simple autour des données d'attribut d'offre ou de liste d'offres.

Contient les zones suivantes :

- *reference* : référence de l'offre ou de la liste d'offres.
- *name* : nom d'offre ou de liste d'offres (*uacName*) ; non garanti unique.
- *description* : description facultative (*uacDescription*).
- *offerCode* : code d'offre (*uacOfferCode*) s'il s'agit d'une offre, ou NULL s'il s'agit d'une liste d'offres. (Non garanti unique.)

---

## WSOfferCodeOrName

Type d'encapsuleur simple autour des données relatives aux codes d'offre ou noms de la liste d'offres.

Contient les zones suivantes :

- *isCode* : valeur booléenne qui indique si la zone *codeOrName* est un code d'offre supposé (true) ou le nom d'une liste d'offres (false).
- *codeOrName* : code d'offre unique (*uacOfferCode*) d'une offre ou nom de la liste d'offres.

---

## WSOfferValidationInfo

Type d'encapsuleur simple autour des informations de validation de l'offre.

Contient les zones suivantes :

- *errorCode* : s'il n'est pas NULL, définit le code d'erreur de validation alphanumérique. Voir la classe IStandardDefinitions pour les codes d'erreur.
- *errorMessage* : message localisé facultatif décrivant l'erreur (le cas échéant).
- *codeOrName* : code d'offre ou nom de la liste d'offres validé.
- *reference* : référence de l'offre ou de la liste d'offres, si elle est valide.

---

## WSOfferTemplateInfo

Type d'encapsuleur simple autour des données de modèle d'offre.

Contient les zones suivantes :

- *reference* : référence du modèle d'offre.
- *name* : nom de modèle d'offre ; garanti unique.
- *description* : description facultative (*uacDescription*).
- *offerTemplateID* : ID de base de données unique du modèle d'offre.

---

## WSBulkOfferInfo

Permet de créer des offres en masse.

Contient les zones suivantes :

- *offerName* : nom de l'offre en cours de création.
- *attributes* : matrice de types WSAtribute qui indique les attributs de l'offre.

---

## WSOfferInfoStatus

Type de retour pour la méthode d'API *bulkCreateOffers()* qui indique le statut de la création d'offre en masse.

Contient les zones suivantes :

- *name* : nom de l'offre.
- *code* : code d'offre. Est null si la création de l'offre échoue.
- *description* : description de l'offre.
- *reference* : WSReference de l'offre créée. Est null si la création de l'offre échoue.
- *status* : instance de WSRequestStatus indiquant le statut de la création d'offre.

---

## Chapitre 3. Exceptions courantes

Cette section décrit un certain nombre d'exceptions courantes émises par l'API CampaignServices.

Tous les messages d'exception localisés apparaissent dans les paramètres régionaux demandés, s'ils sont disponibles pour IBM Campaign. Les règles de rémigration habituelles des paramètres régionaux Java s'appliquent.

---

### RemoteException

**Remarque :** Cet élément s'applique uniquement à l'interface SOAP.

Tous les appels SOAP à l'API peuvent émettre une *RemoteException* en cas d'erreur système, telle qu'un problème dans la couche de traitement de l'enveloppe SOAP (Axis) ou la violation, pour une raison ou pour une autre, d'une contrainte définie dans le langage WSDL du service Web.

Les exceptions d'API vérifiées et non vérifiées ordinaires, telles que *DataException*, sont renvoyées sous la forme d'un statut d'erreur, et non de *RemoteException*.

Consultez la section relative à l'interface SOAP pour plus d'informations.

---

### AuthenticationException

L'utilisateur n'a pas pu être authentifié pour la partition Campaign spécifiée. Vérifiez le rôle utilisateur qui est configuré dans la plateforme marketing.

---

### AuthorizationException

L'utilisateur n'est pas autorisé à effectuer l'opération demandée. Cette exception peut être émise par n'importe quelle méthode API. Elle n'est donc pas déclarée (non vérifiée). Vérifiez les droits affectés au rôle de l'utilisateur dans IBM Marketing Platform.

---

### DataException

Une exception fatale s'est produite dans la couche de base de données sous-jacente dans Campaign (non vérifiée).

Consultez les journaux du programme d'écoute et du diagramme Campaign pour plus de détails.

---

### LockException

Exception temporaire émise lorsque le client tente de mettre à jour un composant, tel qu'un diagramme, alors que celui-ci est en cours de changement par un autre utilisateur. En règle générale, il suffit d'attendre un moment et de renouveler l'opération pour poursuivre après cette exception. Par contre, la logique de relance est de la responsabilité du client.

---

## InvalidComponentException

Une tentative de référencement d'un composant non valide ou inconnu (campagne, diagramme, population ciblée) a été effectuée. La méthode `getComponentReference()` de l'exception renvoie la référence du composant qui pose problème.

---

## InvalidAttributeException

Une exception qui est lancée lorsque le client fournit ou référence un attribut non valide, comme lorsqu'il utilise un type de données incorrect ou une grappe de valeurs alors qu'aucune n'est autorisée. La méthode `getAttributeName()` de l'exception renvoie le nom de l'attribut erroné, `getAttributeValue()` renvoie la valeur et `getComponentReference()` identifie le composant (ou l'index en masse).

---

## AttributeNotFoundException

Emise lorsque le client tente de faire référence à un attribut inconnu (campagne, diagramme, population ciblée, etc.). La méthode `getAttributeName()` de l'exception renvoie le nom de l'attribut sans correspondance ; `getComponentReference()` identifie le composant (ou l'index en masse).

---

## AttributeExistsException

Emise lorsque le client tente de définir les métadonnées d'un attribut en double pour un composant. La méthode `getAttributeName()` de l'exception renvoie le nom de l'attribut en double ; `getComponentReference()` identifie le composant (ou l'index en masse).

---

## CompositeException

Une *CompositeException* est utilisée par certaines API pour signaler plusieurs erreurs à l'appelant. En principe, plusieurs causes sont liées ; elles sont toutes capturées sous forme de liste, dans l'ordre d'apparition. La méthode `getCauseList()` de l'exception renvoie cette liste, qui peut être inspectée pour obtenir des détails sur chaque erreur.

**Remarque :** Généralement, l'API s'exécute ou annule son travail avant d'émettre une exception *CompositeException*. Voir, par exemple, les API de liste des populations ciblées en masse décrites dans «Méthodes de population ciblée», à la page 29.



---

## Chapitre 4. Méthodes d'API CampaignServices

Cette section définit les principales méthodes exportées par l'API CampaignServices 3.0.

---

### Méthodes de service

L'API offre une façon de déterminer les informations d'identification du service même.

#### getServiceInfo

```
WSServiceInfo getServiceInfo()  
    throws CampaignServicesException;
```

Renvoie des informations sur le service, telles que la version la plus récente de l'API prise en charge, la version complète de l'instance IBM Campaign sous-jacente.

**Remarque :** Aucune information client n'est requise par cet appel et aucun droit de sécurité n'est appliqué.

#### Paramètres

Aucun.

#### Renvoie

Renvoie une instance *WSServiceInfo*.

#### Erreurs

Aucune.

---

### Attributs

La plupart des données de l'instance de composant sont affichées par l'API en tant qu'*attributs* ou *métadonnées* d'attribut. Dans certains cas, les définitions de métadonnées d'attribut sont globalement applicables à IBM Campaign (comme les attributs de campagne personnalisés) ; dans d'autres cas, elles se limitent à un composant particulier (comme les variables utilisateur de diagramme).

Sauf indication contraire, tous les attributs peuvent être lus si le client dispose des droits de sécurité nécessaires pour le faire.

**Remarque :** Seuls les composants actifs et auxquels le client a accès sont affichés par cette API. La prise en charge publique se limite aux attributs d'offre, au modèle d'offre et aux métadonnées.

#### getAttributesByName

```
Map<String, WSAttribute>  
    getAttributesByName(String userCredential, String partitionName,  
        Locale requestedLocale,  
        WSReference reference,  
        String[] names)  
    throws CampaignServicesException;
```

Extrait les attributs nommés associés à l'instance de composant spécifiée (peut être vide).

## Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande ; s'il n'est pas indiqué, les préférences de paramètres régionaux de l'utilisateur IBM sont utilisées. L'algorithme de sélection par défaut des paramètres régionaux normaux est appliqué si nécessaire.

*partitionName* : nom facultatif de la partition de campagne à utiliser. S'il n'est pas défini, la partition par défaut est utilisée.

*reference* : référence pour l'instance de composant contenant les attributs souhaités. *InvalidComponentException* est émise si la référence n'est pas valide ou si le composant n'existe pas.

*names* : matrice de noms facultative des attributs à extraire (pas les noms d'affichage) ; si elle n'est pas fournie, tous les attributs sont renvoyés. Emet *AttributeNotFoundException* si l'un des attributs nommés n'existe pas.

## Renvoie

Une mappe typée de zéro ou plusieurs attributs ; le nom de l'attribut est la clé d'entrée de mappe et l'instance d'attribut correspond à la valeur d'entrée.

## Erreurs

*InvalidComponentException*, *AttributeNotFoundException*

*AuthorizationException*, *DataException*

**Remarque :** Tous ces exceptions sont encapsulées dans *CampaignServicesException*.

## updateAttributes

```
void updateAttributes(String userCredential, String partitionName,  
    Locale requestedLocale, WSReference reference,  
    boolean allowCreate,  
    WSAttribute[] attributes)  
    throws CampaignServicesException;
```

Mise à jour d'un ou de plusieurs attributs de l'instance de composant à l'aide des valeurs d'attribut fournies.

## Logique de mise à jour

La logique de mise à jour est la suivante.

Pour chaque attribut contenu dans la mappe d'attributs fournie :

1. Si le nom de l'attribut correspond à un attribut existant, essayez d'écraser sa zone de *valeurs* par la zone de valeurs fournie.
2. Si l'attribut n'existe pas encore, *allowCreate* a la valeur true et ses métadonnées sont connues : créez l'attribut. Cela s'applique aux métadonnées d'attribut global ainsi qu'aux attributs d'instance (à l'exception des diagrammes).
3. Si le type de valeur ou un autre aspect de la définition des métadonnées de l'attribut n'est pas satisfait, ou si une ou plusieurs des valeurs fournies ne sont pas valides, hors plage, lancez une exception *InvalidAttributeException*.

4. Sinon, émettez *AttributeNotFoundException* si l'attribut nommé n'existe pas.

**Remarque :** En cas d'exception, aucune des mises à jour n'est validée.

Cette méthode particulière ne prend pas en charge la définition de nouveaux attributs personnalisés ; pour cela, utilisez la méthode `createAttributeMetadata()`.

Dans tous les cas, l'opération de mise à jour de l'attribut est soumise aux contraintes de sécurité et à la validation habituelles. Le client est chargé de déterminer les attributs requis par une instance de composant particulière, les types corrects, etc.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*reference* : référence pour l'instance de composant contenant les attributs à mettre à jour.

*allowCreate* : indique si un nouvel attribut doit être créé s'il n'existe pas encore pour le composant. (Voir «Logique de mise à jour», à la page 20.)

*attributes* : tableau d'attributs à mettre à jour ; le nom d'attribut est utilisé pour localiser l'attribut à mettre à jour et les nouvelles valeurs sont utilisées pour mettre à jour la valeur de l'attribut existant sous la forme d'un seul objet du type approprié ou d'une matrice, le cas échéant. (Voir Chapitre 3, «Exceptions courantes», à la page 17.)

### Renvoie

Aucune.

### Erreurs

*InvalidComponentException*, *AttributeNotFoundException*,  
*InvalidAttributeException*

*AuthorizationException*, *DataException*

## getAttributeMetadataByName

```
Map<String, WSAttributeMetadata>  
getAttributeMetadataByName(String userCredential,  
    String partitionName, Locale requestedLocale,  
    WSReference reference, String[] names)  
throws CampaignServicesException;
```

Extrait les définitions de métadonnées d'attribut nommé liées à un composant particulier, un modèle, ou définies de manière globale.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*reference* : référence facultative pour le composant ou modèle qui contient les métadonnées d'attribut souhaitées. Si seul `ComponentTypeEnum` est fourni, l'extraction est limitée aux composants de ce type. Si la référence n'est pas fournie, l'extraction renvoie toutes les définitions de métadonnées globales, pour tous les types de composants. Emet *InvalidComponentException* si la référence fournie est non valide.

*names* : matrice de noms facultative des métadonnées d'attribut à récupérer. Si elle n'est pas fournie, toutes les métadonnées sont renvoyées pour le composant, ou définies de manière globale si aucune référence n'est fournie. Emet *AttributeNotFoundException* si une ou plusieurs des définitions de métadonnées d'attribut spécifié n'existent pas.

## Renvoie

Une mappe typée de zéro ou plusieurs définitions de métadonnées d'attribut ; le nom de l'attribut est la clé d'entrée de mappe et les métadonnées de l'attribut correspondent à la valeur d'entrée.

## Erreurs

`InvalidComponentException`, `AttributeNotFoundException`

`AuthorizationException`, `DataException`

## createAttributeMetadata

```
void createAttributeMetadata(String userCredential,  
    String partitionName,  
    Locale requestedLocale, WReference reference,  
    WAttributeMetadata[] attributeMetadata)  
    throws CampaignServicesException;
```

Création d'une ou plusieurs nouvelles définitions de métadonnées d'attribut et, éventuellement, liaison de ces définitions à un composant ou modèle particulier.

## Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*reference* : référence facultative pour le composant ou modèle auquel les métadonnées doivent être liées. Si elle n'est pas fournie, la définition de métadonnées créée est globale. Si la référence est fournie mais n'est pas valide, l'exception *InvalidComponentException* est émise.

*attributeMetadata* : matrice de définitions de métadonnées d'attribut à lier. Si une ou plusieurs des métadonnées spécifiées sont déjà liées au composant, c'est-à-dire si le nom n'est pas unique, l'exception *AttributeExistsException* est émise. L'exception *InvalidAttributeException* est émise s'il existe un problème avec une ou plusieurs des métadonnées spécifiées, c'est-à-dire si elles sont incohérentes en interne.

## Renvoie

Aucune.

## Erreurs

InvalidComponentException, AttributeExistsException, InvalidAttributeException  
AuthorizationException, DataException

## updateAttributeMetadata

```
void updateAttributeMetadata(String userCredential,  
    String partitionName,  
    Locale requestedLocale, WSReference reference,  
    boolean allowCreate,  
    WSAttributeMetadata[] attributeMetadata)  
    throws CampaignServicesException;
```

Mise à jour d'une ou plusieurs définitions de métadonnées d'attribut du composant ou modèle spécifié, avec création éventuelle de nouvelles définitions de métadonnées si nécessaire.

## Logique de mise à jour

La logique de mise à jour est la suivante.

Pour chaque définition de métadonnées d'attribut figurant dans la matrice fournie :

1. Si le nom de l'attribut ne correspond pas à une métadonnée existante liée au composant, effectuez les opérations suivantes selon la valeur du paramètre *allowCreate* :
  - a. *True* : créez une nouvelle définition de métadonnées. Fonctionnellement identique à l'utilisation de la demande `createAttributeMetadata()`.
  - b. *False* : émettez `AttributeNotFoundException`.
2. Si le type de données des métadonnées d'attribut est différent, émettez `InvalidAttributeException`.
3. Essayez d'écraser la définition de métadonnées d'attribut existante par les valeurs de zone des métadonnées fournies, sinon émettez `InvalidAttributeException`. Seules les mises à jour suivantes sont prises en charge (sinon, émettez `InvalidAttributeException`) :
  - a. *name* : non modifiable (name est la clé !).
  - b. *displayName* : acceptez la nouvelle valeur.
  - c. *description* : acceptez la nouvelle valeur.
  - d. *isRequired* : autorisez uniquement le passage de *true* à *false*.
  - e. *isInternal* : acceptez la nouvelle valeur.
  - f. *isGenerated* : aucun changement autorisé.
  - g. *attributeTypeEnum* : aucun changement autorisé
  - h. *accessTypeEnum* : acceptez la nouvelle valeur.
  - i. *selectTypeEnum* : acceptez ces transitions si des options sont fournies :
    - 1) NONE à SINGLE\_SELECT ou MULTIPLE\_SELECT
    - 2) SINGLE\_SELECT à MULTIPLE\_SELECT
  - j. *options* : des options peuvent être ajoutées, mais pas supprimées. Seuls les changements d'options suivantes sont prises en charge (conformément à la correspondance de valeur) :
    - 1) *displayName* : acceptez la nouvelle valeur (pas de propagation).
    - 2) *description* : acceptez la nouvelle valeur (pas de propagation).

- 3) *isDefault* : acceptez la nouvelle valeur ; doit toutefois correspondre à *SelectTypeEnum*.
  - 4) *value* : aucune modification n'est autorisée (la valeur est la clé).
  - k. *defaultValue* (diagrammes uniquement) : acceptez la nouvelle valeur par défaut.
  - l. *maxLength* (texte uniquement) : acceptez la nouvelle longueur si elle est supérieure.
4. Si la définition de métadonnées d'attribut n'est pas cohérente en interne, émettez *InvalidAttributeException*.
  5. Si nécessaire, recherchez toutes les instances de composant qui font référence aux métadonnées d'attribut à jour et effectuez la mise à jour le cas échéant.

**Remarque :** En cas d'exception, aucune des mises à jour n'est validée.

Dans tous les cas, l'opération de mise à jour de l'attribut est soumise aux contraintes de sécurité et à la validation habituelles.

Voir `createAttributeMetadata()`, `deleteAttributeMetadata()`

## Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*reference* : référence facultative pour l'instance de composant qui contient les attributs souhaités. Si elle n'est pas fournie, la mise à jour se limite aux définitions de métadonnées globales. Emet *InvalidComponentException* si la référence fournie est non valide.

*allowCreate* : si sa valeur est true, les définitions de métadonnées qui n'existent pas actuellement sont créées (fonctionnellement équivalent à l'utilisation de la méthode `createAttributeMetadata()`).

*attributeMetadata* : matrice de définitions de métadonnées d'attribut à mettre à jour (et à ajouter si l'indicateur *allowCreate* a la valeur true). Le nom de l'attribut est utilisé pour localiser la définition de métadonnées à mettre à jour et les données restantes servent à mettre à jour la définition existante. (Voir «Logique de mise à jour», à la page 23.)

## Renvoie

Aucune.

## Erreurs

*InvalidComponentException*, *InvalidAttributeException*

*AuthorizationException*, *DataException*

## deleteAttributeMetadata

```
void deleteAttributeMetadata(String userCredential,
    String partitionName,
    Locale requestedLocale, WsReference reference,
    String[] names)
    throws CampaignServicesException;
```

Supprime une ou plusieurs définitions de métadonnées d'attribut nommé du composant spécifié, du modèle (métadonnées d'attribut personnalisé uniquement) ou des définitions de métadonnées d'attribut global.

Dans le cadre de cette tâche, la méthode recherche tous les composants faisant référence aux métadonnées supprimées, et effectue la mise à jour si nécessaire.

**Remarque :** Toutefois, en cas d'exception, aucune des suppressions n'est validée.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*reference* : référence facultative du composant ou modèle qui contient les attributs à supprimer. Si elle n'est pas fournie, la suppression se limite aux définitions de métadonnées globales. Emet *InvalidComponentException* si la référence fournie est non valide.

**Remarque :** Si la matrice de noms facultative des métadonnées d'attribut n'est pas fournie, cette méthode tente de supprimer toutes les métadonnées d'attribut personnalisé associées au composant, ou toutes les définitions globales si la référence n'a pas été fournie.

*names* : matrice de noms facultative des métadonnées d'attribut à supprimer. Emet *AttributeNotFoundException* si une ou plusieurs des métadonnées d'attribut nommé n'existent pas. Emet *InvalidAttributeException* si un attribut n'a pas pu être supprimé.

### Renvoie

Aucune.

### Erreurs

*InvalidComponentException*, *AttributeNotFoundException*,  
*InvalidAttributeException*

*AuthorizationException*, *DataException*

---

## Campagnes

L'API prend en charge les opérations suivantes sur les campagnes (sous réserve des droits de sécurité) :

- création d'une nouvelle campagne
- reconnaissance (liste des campagnes selon différents critères)
- création, lecture et mise à jour d'attribut (via les API d'attribut)

Un certain nombre d'attributs standard sont associés aux campagnes et sont affichés par l'API. Le client peut étendre cette liste à son gré en ajoutant des attributs personnalisés (voir les API d'attributs).

Les attributs de campagne standard sont les suivants :

- *uacName* : nom de campagne (non garanti unique).
- *uacDescription* : chaîne facultative qui décrit la campagne.

- *uacCampaignCode* : code de chaîne identifiant de manière unique la campagne. En principe, il est généré automatiquement par Campaign, mais il peut être fourni par le client.
- *uacCreateDate* : calendrier définissant la date & l'heure de création de la campagne par le serveur.
- *uacUpdateDate* : calendrier indiquant la date et l'heure de la dernière mise à jour de la campagne par le serveur.
- *uacInitiative* : chaîne facultative qui définit l'initiative de campagne.
- *uacObjectives* : chaîne facultative qui identifie les objectifs de la campagne.
- *uacStartDate* : calendrier facultatif indiquant la date & l'heure de lancement (ou de planification du lancement) de la campagne par le serveur.
- *uacEndDate* : comme *uacStartDate*, mais définit la date & l'heure de fin (ou de planification de la fin) de la campagne. Doit être postérieure à *uacStartDate*.
- *uacLastRunDate* : calendrier facultatif indiquant la date & l'heure de la dernière exécution d'un diagramme lié à la campagne (sinon null).
- *uacExternalLinkOwner* : chaîne facultative qui définit le nom du propriétaire d'un lien externe (voir l'attribut *uacExternalLinkReference*). Réservé à IBM ; doit être l'un des éléments suivants :
  - “Plan” (désormais connu sous le nom IBM Marketing Operations)
  - “Collaborate” (désormais connu sous le nom IBM Distributed Marketing)
- *uacExternalLinkId* : identificateur de base de données numérique facultatif qui est affecté par une autre application IBM à un objet lié à cette campagne. Réservé à IBM : voir aussi l'attribut *uacExternalLinkOwner*.

## generateCampaignCode

```
String generateCampaignCode(String userCredential,
    String partitionName,
    Locale requestedLocale);
```

Génération d'un nouveau code de campagne.

Ce code est garanti unique et différent de la valeur renvoyée par un appel précédent ou à venir à cette méthode, ou à la méthode `createCampaign()`, ou de la valeur générée pour une campagne créée par l'interface graphique utilisateur IBM Campaign.

**Remarque :** L'utilisation de cette méthode est facultative, car l'API `createCampaign()` génère un code de campagne pour le client si aucun code n'a été fourni.

Voir `createCampaign()`.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser. S'il existe une seule partition dans l'installation de Campaign, cet argument peut être NULL.

### Renvoie

Le code de campagne généré.



## Erreurs

AuthorizationException, DataException

## deleteCampaigns

```
public WSDDeleteCampaignsResponse deleteCampaigns(String userCredential,  
String partitionName, Locale requestedLocale, WSReference[] wsReferences)  
throws CampaignServicesException
```

Supprime les campagnes spécifiées du système.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*wsReference* : références des campagnes à supprimer.

### Renvois

Renvoie l'objet de type WSDDeleteCampaignsResponse.

### Erreurs

Emet l'exception CampaignServicesException si la campagne n'existe pas ou si la référence n'est pas valide ou si aucune référence n'est fournie.

## createCampaign

```
CampaignInfo createCampaign(String userCredential,  
String partitionName,  
Locale requestedLocale,  
String securityPolicyName,  
String name, Attribute[] attributes)  
throws InvalidFolderException, AttributeNotFoundException,  
InvalidAttributeException;
```

Création d'une nouvelle campagne pour le client, une partition et securityPolicyName, en appliquant les attributs spécifiés. Toutes les campagnes créées par cette API se trouvent sous le dossier racine.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*securityPolicyName* : nom facultatif de la stratégie de sécurité de campagne à utiliser pour créer la campagne. Toutes les opérations ultérieures sur cette campagne utilisent cette stratégie. Si ce paramètre n'est pas défini, la stratégie globale est utilisée.

*name* : nom à affecter à la nouvelle instance de campagne (son attribut "uacName").

*attributes* : matrice facultative des attributs d'initialisation ; les attributs fournis écrasent les valeurs par défaut de la campagne ; les autres ne sont pas modifiées. Par exemple, si un attribut *uacCampaignCode* est fourni, il est utilisé à la place d'un attribut généré automatiquement. C'est au client de déterminer les attributs requis par la campagne, leurs types, etc.

Emet *AttributeNotFoundException* si un ou plusieurs des attributs nommés n'existent pas ou *InvalidAttributeException* si une valeur d'attribut est non valide (par exemple, un type de données incorrect).

## Renvoie

Une seule instance de *CampaignInfo* pour la campagne créée.

## Erreurs

*InvalidAttributeException*, *AttributeNotFoundException*

*AuthorizationException*, *DataException*

## listCampaignsByPage

```
List<CampaignInfo>  
listCampaignsByPage(String userCredential, String partitionName,  
    Locale requestedLocale, Attribute[] attributes,  
    long pageOffset, int pageSize)  
throws AttributeNotFoundException, InvalidAttributeException,  
    RangeException;
```

Énumération d'une "page" de campagnes correspondant aux valeurs d'attribut facultatives, en commençant par le décalage de la page spécifiée. Les dossiers sont ignorés.

Une fois extraite, chaque *CampaignInfo* renvoyée peut être utilisée en l'état, par exemple pour afficher une liste récapitulative, ou bien les méthodes d'attributs peuvent être utilisées pour extraire ou mettre à jour les attributs de campagne.

Cette API ne gère aucun état. Il est donc possible de l'appeler dans n'importe quel ordre.

## Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*attributes* : tableau facultatif des attributs pour correspondance ; le nom de l'attribut, le type de données et les valeurs sont utilisés pour déterminer la correspondance ; si l'attribut prend en charge les matrices, toutes les valeurs spécifiées doivent correspondre. L'opérateur de correspondance implicite est ET. Par conséquent, seules les campagnes correspondant à toutes les valeurs d'attribut fournies sont renvoyées.

Emet *AttributeNotFoundException* si un nom d'attribut n'existe pas ou *InvalidAttributeException* si un ou plusieurs des attributs fournis sont non valides.

*pageOffset* : décalage de début de toutes les campagnes possibles pour commencer l'énumération (valeur zéro). Par exemple, si l'énumération correspond à 1000

campagnes et si cette valeur est définie sur 10, la page commencera au 11ème composant. Une *RangeException* est émise si le décalage fourni est hors plage.

*pageSize* : nombre maximum de campagnes correspondantes à renvoyer pour la page (ne doit pas dépasser 500).

### Renvoie

Une liste typée de zéro ou plusieurs instances d'encapsuleur de données *CampaignInfo*, une pour chaque campagne correspondante dans la page.

### Erreurs

*AttributeNotFoundException*, *InvalidAttributeException*, *RangeException*

*InvalidExecutionContextException*, *AuthorizationException*

---

## Méthodes de population ciblée

Les populations ciblées sont une abstraction de certains sous-ensembles connus de résultats de campagne et sont gérées par IBM Campaign en tant que liste des populations ciblées (LPC). Les populations ciblées peuvent concerner une campagne de manière globale ou être associées à un diagramme de campagne particulier.

L'API prend en charge les opérations suivantes sur les populations ciblées :

- création d'une ou plusieurs nouvelles populations ciblées globales ;
- mise à jour en bloc d'une ou plusieurs populations ciblées existantes ;
- reconnaissance (liste des populations ciblées) ;
- création, lecture et mise à jour d'attribut (via les API d'attribut) ;
- suppression d'une population ciblée existante ;
- extrait des résultats d'exécution associés à une ou plusieurs cibles.

Un certain nombre d'attributs standard sont associés aux populations ciblées et sont affichés par l'API. Le client peut étendre cette liste à son gré en ajoutant des définitions de métadonnées d'attributs personnalisés (voir les API d'attributs). Chaque métadonnée d'attribut peut être considérée comme une colonne dans la LPC ; la présentation de la liste appartient au client.

Les attributs de population ciblée standard sont les suivants :

- *uacName* : nom de la cible.
- *uacDescription* : chaîne facultative qui décrit le diagramme.
- *uacCellCode* : chaîne de code identifiant la cible de manière unique. En principe, il est généré automatiquement par Campaign, mais il peut être fourni par le client.
- *uacCreateDate* : instance de calendrier qui indique la date & l'heure de création de la cible par le serveur.
- *uacUpdateDate* : instance de calendrier qui définit le moment de la dernière mise à jour de la cible par le serveur.
- *uacIsControl* : valeur booléenne qui indique s'il s'agit d'une cible de contrôle (true) ou non (false). D'autres cibles peuvent désigner cette cible comme cible de contrôle (voir *uacControlCell*).
- *uacControlCell* : référence facultative de la cible de contrôle (non autorisée s'il s'agit d'une cible de contrôle). Voir l'attribut *uaclsControl*.

- *uacIsApproved* : valeur booléenne qui indique si la cible est validée (true) ou non (false).
- *uacIsReadOnly* : valeur booléenne qui indique si la cible est en lecture seule (true) ou non (false).
- *uacDisplayOrder* : entier qui indique l'ordre de cette cible (ligne) par rapport aux autres dans la liste des populations ciblées.
- *uacIsTopDown* : valeur booléenne qui indique si la cible est descendante.
- *uacAssignedOffers* : matrice facultative d'une ou plusieurs références d'offres ou listes d'offres affectées à cette cible (non autorisée s'il s'agit d'une cible de contrôle).
- *uacFlowchartName* : nom facultatif du diagramme auquel cette cible est liée (en lecture seule - doit être défini par le biais de l'interface graphique IBM Campaign ; non autorisé s'il s'agit d'une cible de contrôle).
- *uacFlowchartId* : identificateur de base de données facultatif pour le diagramme auquel cette cible est liée (en lecture seule, comme ci-dessus).

## createTargetCell

```
TargetCellInfo
    createTargetCell(String userCredential, String partitionName,
        Locale requestedLocale,
        Reference campaignReference,
        Attribute[] attributes)
        throws InvalidComponentException, CompositeException;
```

Création d'une nouvelle ligne de population ciblée propre à la campagne, avec application des informations utilisateur et des attributs spécifiés par cible.

Les attributs spécifiés peuvent être standard ou personnalisés ; toutefois, s'ils sont personnalisés, les définitions de métadonnées d'attribut global correspondantes doivent exister.

Une fois la population ciblée créée, les valeurs d'attribut peuvent être changées à l'aide des API d'attributs.

Voir `listTargetCells()`, `bulkCreateTargetCells()`.

Voir `createAttributeMetadata()`, `listAttributeMetadata()`, `getAttributesByName()`

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*campaignReference* : référence de la campagne qui contient la liste des populations ciblées à mettre à jour. Cumule une exception *InvalidComponentException* si la campagne n'existe pas ou si la référence n'est pas valide.

*attributes* : matrice facultative d'attributs LPC pour la nouvelle cible. Chaque élément d'attribut fourni écrase les valeurs par défaut de l'attribut de cible correspondant ; les autres ne sont pas modifiées. C'est au client de déterminer les attributs requis par la cible, leurs types, etc. Cumule une exception *InvalidAttributeException* en cas de problème avec un attribut spécifié.

Si des exceptions sont accumulées, cette méthode lance *CompositeException* et toutes les créations sont annulées. La liste des causes de l'exception inclut une exception pour chaque attribut à l'origine de l'erreur et comprend un index numérique à la place de la *référence*, le nom de l'attribut et généralement la valeur qui pose problème. La liste des causes est triée comme avec l'entrée *attributeList*.

### Retour

Un encapsuleur de données *TargetCellInfo* pour la cible LPC créée.

### Erreurs

*InvalidComponentException*, *CompositeException*

*AuthorizationException*, *DataException*

## bulkCreateTargetCells

```
List<TargetCellInfo>  
    bulkCreateTargetCells(String userCredential,  
        String partitionName,  
        Locale requestedLocale,  
        Reference campaignReference,  
        List<Attribute[]> attributesList)  
    throws InvalidComponentException, CompositeException;
```

Création de plusieurs lignes de population ciblée propres à la campagne à la fois, avec application des informations utilisateur et des attributs spécifiés par cible.

Les attributs spécifiés peuvent être standard ou personnalisés ; toutefois, s'ils sont personnalisés, les définitions de métadonnées d'attribut global correspondantes doivent exister.

Une fois la population ciblée créée, les valeurs d'attribut peuvent être changées à l'aide des API d'attributs.

Voir `listTargetCells()`.

Voir `createAttributeMetadata()`, `listAttributeMetadata()`, `getAttributesByName()`

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*campaignReference* : référence de la campagne qui contient la liste des populations ciblées à mettre à jour. Cumule une exception *InvalidComponentException* si la campagne n'existe pas ou si la référence n'est pas valide.

*attributeList* : liste facultative des matrices d'attributs par cible, un pour chaque ligne de population ciblée à créer. Les attributs fournis pour un élément de liste particulier écrasent les valeurs par défaut de l'attribut de cible correspondant ; les autres ne sont pas modifiées. C'est au client de déterminer les attributs requis par la cible, leurs types, etc. Cumule une exception *InvalidAttributeException* en cas de problème avec un attribut spécifié.

Si des exceptions sont cumulées, cette méthode lance *CompositeException* et toutes les créations sont annulées. La liste des causes de l'exception inclut une exception

pour chaque attribut à l'origine de l'erreur et comprend un index numérique à la place de la *référence*, et le nom de l'attribut, etc. La liste des causes est triée comme avec l'entrée *attributeList*.

### **Renvoie**

Une liste d'encapsuleurs de données *TargetCellInfo* pour chaque instance créée, classés en fonction de l'ordre des éléments du paramètre *attributesList* d'entrée.

### **Erreurs**

*InvalidComponentException*, *CompositeException*

*AuthorizationException*, *DataException*

## **listTargetCells**

```
List<TargetCellInfo>  
listTargetCells(String userCredential,  
Reference campaignReference, Locale requestedLocale,  
Attribute[] attributes)  
throws InvalidComponentException, InvalidAttributeException;
```

Répertorie les informations relatives à toutes les populations ciblées existant actuellement et correspondant aux attributs spécifiés, soit pour la campagne spécifiée soit de manière globale si aucune campagne n'est spécifiée.

Voir *getAttributeMetadata()*, *getAttributesByName()*.

### **Paramètres**

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*campaignReference* : référence de la campagne parente. Emet une exception *InvalidComponentException* si la campagne n'existe pas ou si la référence n'est pas valide.

*attributes* : matrice facultative des attributs de correspondance. L'opérateur de correspondance implicite est AND. Par conséquent, seules les cibles correspondant à toutes les valeurs d'attribut fournies sont renvoyées.

Emet *InvalidAttributeException* si un ou plusieurs des attributs spécifiés sont non valides.

### **Renvoie**

Renvoie une liste de zéro ou plusieurs instances *TargetCellInfo* pour les cibles correspondantes.

### **Erreurs**

*InvalidComponentException*, *InvalidAttributeException*

*AuthorizationException*, *DataException*

## bulkUpdateTargetCells

```
void bulkUpdateTargetCells(String userCredential,  
    String partitionName,  
    Locale requestedLocale,  
    Map<Reference, Attribute[]> attributesMap)  
    throws CompositeException;
```

Mise à jour des attributs d'une ou plusieurs populations ciblées.

La logique de mise à jour est la suivante.

Pour chaque élément du *attributesMap* fourni, la clé d'entrée est la référence de la population ciblée à mettre à jour et la valeur d'entrée est une matrice d'attributs de mise à jour pour cette cible. Si la population ciblée n'existe pas, cumulez une exception *InvalidComponentException*.

Une fois la population ciblée localisée, pour chaque attribut spécifié, procédez comme suit :

1. Si le nom de l'attribut correspond à un attribut existant, essayez d'écraser sa zone de valeurs par la zone de valeurs fournie.
2. Si le type de valeur ou un autre aspect de la définition des métadonnées de l'attribut n'est pas satisfait, ou si une ou plusieurs des valeurs fournies ne sont pas valides, hors plage, cumulez une exception *InvalidAttributeException*.
3. Sinon cumulez *AttributeNotFoundException* si l'attribut nommé n'existe pas.

Si des exceptions sont cumulées, cette méthode lance *CompositeException* et toutes les mises à jour sont annulées. La liste des causes de l'exception contient les exceptions énumérées ci-dessus. Pour chaque attribut à l'origine de l'erreur, la référence et le nom de l'attribut sont enregistrés.

Dans tous les cas, l'opération de mise à jour de l'attribut est soumise aux contraintes de sécurité et à la validation habituelles. Le client est chargé de déterminer les attributs requis par une instance de composant particulière, les types corrects, etc.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*attributesMap* : mappe des populations ciblées à mettre à jour ; la clé d'entrée est la référence de la cible à mettre à jour et la valeur d'entrée est une matrice d'attributs de mise à jour. Le nom d'attribut est utilisé pour localiser l'attribut à mettre à jour et les nouvelles valeurs d'attributs sont utilisées pour mettre à jour la valeur de l'attribut existant sous la forme d'un seul objet du type approprié ou d'une matrice, le cas échéant. Voir les exceptions ci-dessus.

### Renvoie

Aucune.

### Erreurs

ComponentException

AuthorizationException, DataException

## getRunResultsByCell

```
List<RunResults>  
getRunResultsByCell(String userCredential, String partitionName,  
    Locale requestedLocale,  
    Reference[] cellReferences)  
throws InvalidComponentException;
```

Obtention des résultats d'exécution d'une ou plusieurs populations ciblées, éventuellement pour un diagramme n'ayant jamais démarré ou qui est toujours en cours.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*cellReferences* : matrice de références des populations ciblées dont vous souhaitez obtenir les résultats d'exécution. Emet *InvalidComponentException* si une ou plusieurs des références de cibles ne sont pas valides ou font référence à une cible inexistante.

### Renvoie

Renvoie une liste typée des résultats d'exécution pour les cibles nommées, triées en fonction de la matrice de référence d'entrée.

Chaque statut d'exécution est RUNNING si la zone de processus de diagramme sous-jacente est toujours en cours d'exécution, FAILED si l'exécution a échoué pour une raison ou une autre, ou NOT\_STARTED si l'exécution de la zone de processus n'a pas commencé. Des détails sur le statut sont également fournis.

### Erreurs

InvalidComponentException

AuthorizationException, DataException

## bulkDeleteTargetCells

```
void bulkDeleteTargetCells(String userCredential,  
    String partitionName,  
    Locale requestedLocale,  
    Reference[] cellReferences)  
throws CompositeException;
```

Supprime une ou plusieurs populations ciblées existantes et tous leurs composants dépendants (par exemple, liaisons de diagramme, attributs).

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*cellReferences* : matrice ou une ou plusieurs références de cibles à supprimer. *InvalidComponentException* est cumulée s'il existe un problème avec l'une des références spécifiées ou si une cible n'existe pas.



Si des exceptions sont cumulées, cette méthode lance *CompositeException* et toutes les suppressions sont annulées. La liste des causes de l'exception contient les exceptions énumérées ci-dessus. Pour chaque cible à l'origine de l'erreur, la référence est enregistrée.

### **Renvoie**

Aucune.

### **Erreurs**

*CompositeException*

*AuthorizationException*, *DataException*

## **updateTemplateAttributes**

```
updateTemplateAttributes  
(String userCredential, String partitionName, Locale requestedLocale,  
WSReference wsReference, boolean allowCreate,  
boolean clearExisting, WSAttribute[] wsStaticAttributes,  
WSAttribute[] wsHiddenAttributes, WSAttribute[] wsParametricAttributes)  
throws CampaignServicesException
```

Met à jour les attributs des modèles spécifiés

### **Paramètres**

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*wsCampaignReference* : référence de la campagne parent.

*allowCreate* : non utilisé actuellement.

*clearExisting* : si cet indicateur a la valeur true, toutes les valeurs antérieures dans le modèle qui ne sont pas envoyées dans la requête sont effacées.

*wsStaticAttributes* : liste des attributs statiques dans le modèle.

*wsHiddenAttributes* : liste des attributs masqués dans le modèle.

*wsParametricAttributes* : liste des attributs paramétriques dans le modèle.

### **Renvoie**

Aucune.

### **Erreurs**

Emet l'exception *CampaignServicesException* si le modèle d'offre n'existe pas ou si la référence n'est pas valide ou si aucune référence n'est fournie.

## listBottomUpTargetCells

```
public List <WSTargetCellDetails>  
listBottomUpTargetCells(String userCredential, String partitionName,  
Locale requestedLocale, WSReference wsCampaignReference)  
throws CampaignServicesException
```

Répertorie des informations sur toutes les populations ciblées du haut ou du bas qui existent pour la campagne spécifiée.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*wsCampaignReference* : référence de la campagne parent.

### Renvoie

Renvoie une liste de zéro ou plusieurs instances *WSTargetCellDetails* pour les cibles correspondantes

### Erreurs

Emet l'exception *CampaignServicesException* si la campagne n'existe pas ou si la référence n'est pas valide.

---

## Analyse

L'API prend en charge l'extraction d'indicateurs simples à partir d'IBM Campaign.

## getCampaignMetrics

```
MetricsInfo getCampaignMetrics(String userCredential,  
String partitionName,  
Locale requestedLocale,  
Reference campaignReference)  
throws InvalidComponentException;
```

Extrait les indicateurs pour la campagne spécifiée.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*campaignReference* : référence de la campagne parente. Emet l'exception *InvalidComponentException* s'il existe un problème avec la référence de campagne ou si la campagne n'existe pas.

### Renvoie

Renvoie une instance *MetricsInfo* pour la campagne.

## Erreurs

InvalidComponentException

AuthorizationException, DataException

---

## Offre, méthodes de liste d'offres et de modèle d'offre

L'API prend en charge les opérations suivantes sur les offres :

- reconnaissance (liste par dossier (offres, listes d'offres et sous-dossiers), attribut (offres et modèles d'offre) ou valeur de recherche (offres) ;
- validation ;
- extraction d'informations (extraction d'attributs pour une offre ou un modèle d'offre spécifique) ;
- création, changement, retrait et suppression d'offres.

Un certain nombre d'attributs standard sont associés aux offres. Le client peut étendre cette liste à son gré en ajoutant des définitions de métadonnées d'attributs personnalisés (voir les API d'attributs).

Les attributs d'offre standard sont les suivants :

- *uacName* : nom de l'offre.
- *uacDescription* : chaîne facultative qui décrit l'offre.
- *uacOfferCode* : chaîne de code identifiant l'offre de manière unique. En principe, il est généré automatiquement par IBM Campaign, mais il peut être fourni par le client.
- *uacCreateDate* : instance de calendrier qui indique la date & l'heure de création de l'offre par le serveur.
- *uacUpdateDate* : instance de calendrier qui définit le moment de la dernière mise à jour de l'offre par le serveur.

Les modèles d'offre ont également des attributs standard et personnalisés. Les attributs de modèle d'offre standard sont les suivants :

- *uacName* : nom du modèle d'offre.
- *uacDescription* : chaîne facultative qui décrit le modèle d'offre.
- *uacCreateDate* : instance de calendrier qui indique la date & l'heure de création du modèle d'offre par le serveur.
- *uacUpdateDate* : instance de calendrier qui définit le moment de la dernière mise à jour du modèle d'offre par le serveur.

## listOffersAndFolders

```
List<WComponentOrFolderInfo>  
listOffersAndFolders(String userCredential, String partitionName,  
    Locale requestedLocale,  
    WSReference parentReference)  
throws CampaignServicesException;
```

Liste de toutes les offres, listes d'offres et dossiers sous le dossier parent facultatif.

Une fois extraite, chaque instance *WComponentOrFolderInfo* renvoyée peut être utilisée en l'état, par exemple pour afficher le niveau suivant de la hiérarchie de dossiers, les API d'attribut peuvent être utilisées pour mettre à jour les offres contenues.

## Paramètre

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*parentReference* : référence facultative du dossier parent à répertorier. Seules les offres enfant immédiat, les listes d'offres et les dossiers de ce dossier parent sont énumérés ; il est donc nécessaire d'effectuer des appels successifs vers cette API pour parcourir l'arborescence de dossiers complète (mais elle est généralement très superficielle). Si aucun parent n'est indiqué, tous les composants et dossiers situés sous la racine sont renvoyés.

Emet *InvalidFolderException* s'il existe un problème avec la référence de dossier parent spécifiée.

Une liste typée de zéro ou plusieurs instances d'encapsuleur de données *WSComponentOrFolderInfo*, une pour chaque composant ou dossier correspondant.

## Erreurs

*InvalidFolderException*

*InvalidExecutionContextException*, *AuthorizationException*

## searchOffersBasic

```
List<WSOfferInfo>  
  searchOffersBasic(String userCredential, Locale requestedLocale,  
    String partitionName, long folderID,  
    String searchCriteria, boolean includeRetired,  
    int pageOffset, int pageSize)  
  throws CampaignServicesException;
```

Énumération d'une "page" d'offres contenant les critères de recherche indiqués dans les zones Nom, Description, Auteur de création ou Code d'offre, en commençant par le décalage de la page spécifiée. La recherche est basée sur l'entrée de dossier facultative. (Si la valeur 0 est fournie pour l'ID de dossier, le dossier d'offre racine est utilisé par défaut). Les correspondances sont renvoyées sur la base d'une correspondance "contient" pour la chaîne de recherche.

Une fois extraite, chaque *WSOfferInfo* renvoyée peut être utilisée en l'état, par exemple pour afficher une liste récapitulative, ou bien les méthodes d'attributs peuvent être utilisées pour extraire ou mettre à jour les attributs de l'offre.

Cette API ne gère aucun état. Il est donc possible de l'appeler dans n'importe quel ordre.

## Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*folderID* : ID du dossier Offre faisant l'objet de la recherche ; si la valeur 0 est spécifiée pour l'ID de dossier, la recherche porte sur le dossier racine.

*searchCriteria* : phrase de recherche.

*includeRetired* : valeur booléenne indiquant si les résultats de la recherche incluent les offres retirées. Les valeurs possibles sont TRUE et FALSE, TRUE indiquant que les offres retirées sont incluses et FALSE indiquant qu'elles ne le sont pas.

*pageOffset* : décalage de début de tous les composants possibles pour commencer l'énumération (valeur zéro). Par exemple, si l'énumération correspond à 1000 offres et si cette valeur est définie sur 10, la page commencera au 11ème composant. Une RangeException est émise si le décalage fourni est hors page.

*pageSize* : nombre maximum de composants correspondants à renvoyer pour la page (ne doit pas dépasser 500).

## Renvoie

Renvoie une liste typée de zéro ou plusieurs instances d'encapsuleur de données *Offer*, une pour chaque offre renvoyée dans la page.

## Erreurs

RangeException

## listOffersByPage

```
List<OfferInfo>  
listOffersByPage(String userCredential, String partitionName,  
                 Locale requestedLocale, Attribute[] attributes,  
                 long pageOffset, int pageSize)  
throws AttributeNotFoundException, InvalidAttributeException,  
        RangeException;
```

Énumération d'une "page" d'offres correspondant aux valeurs d'attribut facultatives, en commençant par le décalage de la page spécifiée. Les dossiers sont ignorés. Les correspondances sont renvoyées sur la base d'une correspondance "similaire" pour les chaînes (où la correspondance est considérée comme suffisante si une chaîne contient la valeur demandée) et d'une correspondance exacte pour les dates et les nombres.

Une fois extraite, chaque *OfferInfo* renvoyée peut être utilisée en l'état, par exemple pour afficher une liste récapitulative, ou bien les méthodes d'attributs peuvent être utilisées pour extraire ou mettre à jour les attributs de l'offre.

Cette API ne gère aucun état. Il est donc possible de l'appeler dans n'importe quel ordre.

## Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*attributes* : tableau facultatif des attributs pour correspondance ; le nom de l'attribut, le type de données et les valeurs sont utilisés pour déterminer la correspondance ; si l'attribut prend en charge les matrices, toutes les valeurs spécifiées doivent correspondre. L'opérateur de correspondance implicite est OU. Par conséquent, les composants correspondant à l'une des valeurs d'attribut fournies sont renvoyés.

Emet *AttributeNotFoundException* si un nom d'attribut n'existe pas ou *InvalidAttributeException* si un ou plusieurs des attributs fournis sont non valides.

*pageOffset* : décalage de début de tous les composants possibles pour commencer l'énumération (valeur zéro). Par exemple, si l'énumération correspond à 1000 offres et si cette valeur est définie sur 10, la page commencera au 11ème composant. Une *RangeException* est émise si le décalage fourni est hors plage.

*pageSize* : nombre maximum de composants correspondants à renvoyer pour la page (ne doit pas dépasser 500).

### **Renvoie**

Une liste typée de zéro ou plusieurs instances d'encapsuleur de données *OfferInfo*, une pour chaque composant correspondant dans la page.

### **Erreurs**

*AttributeNotFoundException*, *InvalidAttributeException*, *RangeException*

*InvalidExecutionContextException*, *AuthorizationException*

## **createSmartOfferList**

```
public WSCreateSmartOfferListResponse createSmartOfferList
(String userCredential, String partitionName, Locale requestedLocale,
String name, String description,String policyName,
WSReference parentFolder,WSSmartListInfo offerListInfo,
WSApplicationTypeEnum createdBy,long creatorObjectId)
throws CampaignServicesException
```

Crée une liste d'offres dynamiques.

### **Paramètres**

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*name* : nom du dossier à créer.

*description* : description du nouveau dossier.

*securityPolicyName* : nom de la stratégie de sécurité à utiliser.

*parentFolder* : id du dossier où la liste d'offres doit être créée.

*offerListInfo* : objet de type *WSSmartListInfo*.

*createdBy* : (facultatif) l'objet de type *WSApplicationTypeEnum* indique l'application qui a créé le dossier. Valeurs possibles : Campaign/Plan/Collaborate/eMessage. S'il n'est pas fourni, Campaign est utilisé.

*creatorObjectId* : (facultatif) utilisé par le plan pour lier le dossier dans la campagne au dossier dans le plan.

## Renvoie

Renvoie l'objet de type `WSCreateSmartOfferListResponse`.

## Erreurs

Emet l'exception `CampaignServicesException` si l'ID `parentFolder` n'est pas valide ou si `offerListInfo` n'est pas fourni.

Emet l'exception `CampaignServicesException` si le nom de liste est en double.

## `createStaticOfferList`

```
public WSCreateStaticOfferListResponse createStaticOfferList
(String userCredential, String partitionName, Locale requestedLocale, String name,
String description, String policyName, WSReference parentFolder, WSReference[]
listMembers, WSApplicationTypeEnum createdBy, long creatorObjectId)
throws CampaignServicesException
```

Crée une liste d'offres statiques.

## Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*name* : nom du dossier à créer.

*description* : description du nouveau dossier.

*securityPolicyName* : nom de la stratégie de sécurité à utiliser.

*parentFolder* : id du dossier où la liste d'offres doit être créée.

*listMembers* : références aux offres à inclure dans la liste d'offres.

*createdBy* : (facultatif) l'objet de type `WSApplicationTypeEnum` indique l'application qui a créé le dossier. Valeurs possibles : `Campaign/Plan/Collaborate/eMessage`. S'il n'est pas fourni, `Campaign` est utilisé.

*creatorObjectId* : (facultatif) utilisé par le plan pour lier le dossier dans la campagne au dossier dans le plan.

## Renvoie

Renvoie l'objet de type `WSCreateStaticOfferListResponse`

## Erreurs

Emet l'exception `CampaignServicesException` si l'ID `parentFolder` n'est pas valide ou si `listMembers` ne sont pas valides

Emet l'exception `CampaignServicesException` si le nom de liste est en double.

## getOffers

```
public WSGetOffersResponse getOffers  
(String userCredential, String partitionName,  
Locale requestedLocale, WSReference[] wsReferences)  
throws CampaignServicesException
```

Répertorie les détails des offres comme indiqué dans la demande.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*wsCampaignReference* : référence de la campagne parent.

### Renvois

Renvoie l'objet de type WSGetOffersResponse.

### Erreurs

Emet l'exception CampaignServicesException si l'offre n'existe pas ou si la référence n'est pas valide ou si aucune référence n'est fournie.

## validateOffers

```
List<OfferValidationInfo>  
validateOffers(String userCredential, String partitionName,  
Locale requestedLocale,  
OfferCodeOrName[] codeOrNames);
```

Validation des codes d'offre ou des noms de la liste d'offres fournis et renvoi d'informations de validation pour chacun d'eux. La "validation" consiste à vérifier qu'il existe bien une seule et unique offre ou liste d'offres correspondante dans la base de données.

L'objet OfferValidationInfo contient un message d'erreur au lieu de OfferInfo si aucune offre ou liste d'offres ne correspond au code ou nom donné. Vous obtenez également une erreur plutôt qu'une correspondance si le code ou le nom donné correspond à plusieurs offres ou listes d'offres. La liste est renvoyée dans le même ordre qu'indiqué. Les codes d'offre et les noms de la liste d'offres sont validés sur la base d'une correspondance exacte avec les offres.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*codeOrNames* : matrice de tous les codes d'offre ou noms de la liste d'offres à valider.



**Remarque :** Aucune exception n'est émise par cette méthode ; au lieu de cela, des informations de validation sont renvoyées pour tous les codes ou noms fournis.

### **Renvois**

Une liste typée de zéro ou plusieurs instances d'encapsuleur de données *OfferValidationInfo*.

### **Erreurs**

Aucune.

## **editOfferList**

```
public WSEditOfferListResponse editOfferList(String userCredential,  
String partitionName, Locale requestedLocale, WSReference listReference,  
boolean isSmartList, String name, String description,  
WSReference[] listMembers, WSSmartListInfo offerListInfo,  
Long creatorObjectId, boolean clearExisting)  
throws CampaignServicesException
```

Met à jour les détails de liste d'offres dynamiques et statiques.

### **Paramètres**

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*listReference* : référence à la liste d'offres.

*isSmartList* : l'indicateur indique si la liste est dynamique ou statique.

*name* : nom du dossier à créer.

*description* : description du nouveau dossier.

*listMembers* : références aux offres à inclure dans la liste d'offres.

*offerListInfo* : objet de type *WSSmartListInfo*.

*creatorObjectId* : (facultatif) utilisé par le plan pour lier le dossier dans la campagne au dossier dans le plan.

*clearExisting* : l'indicateur indique si des informations existantes doivent être effacées. S'il a la valeur *true*, les membres de liste existants sont effacés avant l'ajout de nouveaux membres. S'il a la valeur *false*, de nouveaux membres sont ajoutés à des membres existants.

### **Renvois**

Renvoie l'objet de type *WSEditOfferListResponse*.

### **Erreurs**

Emet l'exception *CampaignServicesException* si l'ID *parentFolder* n'est pas valide ou si *offerListInfo* n'est pas fourni ou si *listMembers* ne sont pas valides.

Emet l'exception `CampaignServicesException` si le nom de liste est en double.

## createOffer

```
OfferInfo createOffer(String userCredential, String partitionName,
    Locale requestedLocale,
    String securityPolicyName,
    String name, String templateName,
    Attribute[] attributes)
    throws InvalidFolderException, AttributeNotFoundException,
    InvalidAttributeException;

public WSOfferInfo createOffer(String authorizationLoginName, String
    partitionName, Locale requestedLocale, String
    securityPolicyName, String name, long folderID,
    String templateName, WSAttribute[] wsAttributes)
    throws CampaignServicesException;
```

Création d'une nouvelle offre pour le client, en appliquant les attributs spécifiés.

### Paramètres

*authorizationLoginName* : nom d'utilisateur du créateur de l'offre. Les utilisateurs doivent se voir accorder les droits d'ajout d'offres pour utiliser cette méthode.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*securityPolicyName* : nom facultatif de la stratégie de sécurité de campagne à utiliser pour créer l'offre. Toutes les opérations ultérieures sur cette offre utilisent cette stratégie. Si ce paramètre n'est pas défini, la stratégie *globale* est utilisée.

*name* : nom à affecter à la nouvelle instance d'offre (son attribut *uacName*).

*folderID* : ID du dossier Offre où est créée l'offre. L'exactitude de cet ID est validée et une exception est émise si l'ID n'est pas valide.

*templateName* : nom (unique) requis d'un modèle d'offre existant à utiliser pour la nouvelle offre.

*wsAttributes* : matrice des attributs d'initialisation ; les attributs fournis écrasent les valeurs par défaut de l'offre ; les autres ne sont pas modifiées. Par exemple, si un attribut *uacOfferCode* est fourni, il est utilisé à la place d'un attribut généré automatiquement. C'est au client de déterminer les attributs requis par l'offre, leurs types, etc.

Emet `CampaignServicesException` si l'une des conditions suivantes se présente :

- Le paramètre `folderID` est non valide (inexistant ou pas de type offre).
- L'utilisateur n'est pas autorisé à effectuer cette opération.
- Des attributs non valides sont fournis dans `wsAttributes`.
- D'autres exceptions d'exécution se produisent.

### Retour

Une seule instance de `OfferInfo` pour l'offre créée.

### Erreurs

`CampaignServicesException`

## retireOffers

```
void retireOffers(String userCredential, String partitionName,  
                 Locale requestedLocale, WSReference[] references)  
    throws CampaignServicesException;
```

Retire une ou plusieurs offres existantes.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*references* : matrice de références des offres à retirer. *InvalidComponentException* est émise s'il existe un problème avec une référence particulière ou si une offre n'existe pas.

### Renvois

Aucune.

### Erreurs

*InvalidComponentException*

*AuthorizationException*, *DataException*

## deleteOffers

```
void deleteOffers(String userCredential, String partitionName,  
                 Locale requestedLocale, WSReference[] references)  
    throws CampaignServicesException;
```

Supprime une ou plusieurs offres existantes.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*reference* : matrice de références des offres à supprimer. *InvalidComponentException* est émise s'il existe un problème avec une référence spécifiée ou si une offre n'existe pas.

### Renvois

Aucune.

### Erreurs

*InvalidComponentException*

*AuthorizationException*, *DataException*

## deleteOffersAndLists

```
public WSDeleteOffersAndListsResponse deleteOffersAndLists  
(String userCredential, String partitionName, Locale requestedLocale,  
WSReference[] offers)  
throws CampaignServicesException
```

Supprime les offres et les listes spécifiées.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*offers* : grappe de références d'offre ou de liste d'offres.

### Renvoie

Renvoie l'objet de type `WSGetOfferListMembersResponse`.

### Erreurs

Emet l'exception `CampaignServicesException` si l'ID de l'offre ou l'ID de liste d'offres n'est pas valide.

## listOfferTemplates

```
List<WSOfferTemplateInfo>  
listOfferTemplates(String userCredential, String partitionName,  
Locale requestedLocale)  
throws CampaignServicesException;
```

Liste de tous les modèles d'offre que l'utilisateur est autorisé à visualiser.

Une fois extraite, chaque instance `WSOfferTemplateInfo` renvoyée peut être utilisée en l'état, ou une ou plusieurs des API d'attribut peuvent être utilisées pour extraire ou mettre à jour un modèle répertorié.

### Paramètre

*userCredential* : données d'identification de l'utilisateur client.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

### Renvoie

Une liste typée de zéro ou plusieurs instances d'encapsuleur de données `WSOfferTemplateInfo`, une pour chaque modèle renvoyé.

### Erreurs

`InvalidExecutionContextException`, `AuthorizationException`

`DataException`

## createTemplate

```
createTemplate(String userCredential, String partitionName, Locale requestedLocale,  
String name, String securityPolicyName, WSAttribute[]  
wsStaticAttributes, WSAttribute[] wsHiddenAttributes,  
WSAttribute[] wsParametricAttributes)  
throws CampaignServicesException
```

Crée un modèle d'offre.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*name* : nom du nouveau modèle d'offre.

*securityPolicyName* : nom de la stratégie de sécurité à utiliser.

*wsStaticAttributes* : liste des attributs statiques dans le modèle.

*wsHiddenAttributes* : liste des attributs masqués dans le modèle.

*wsParametricAttributes* : liste des attributs paramétriques dans le modèle.

### Renvoie

Renvoie l'objet de type `WSCreateTemplateResponse`.

### Erreurs

Emet l'exception `CampaignServicesException` si le modèle d'offre n'existe pas ou si la référence n'est pas valide ou si aucune référence n'est fournie.

## getOfferTemplate

```
public WSGetOfferTemplateResponse getOfferTemplate(String userCredential,  
String partitionName, Locale requestedLocale, WSReference[] wsReferences)  
throws CampaignServicesException
```

Répertorie les détails des modèles d'offre tels qu'indiqués dans les références.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*wsCampaignReference* : référence de la campagne parent.

### Renvoie

Renvoie l'objet de type `WSGetOfferTemplateResponse`.

## Erreurs

Emet l'exception CampaignServicesException si le modèle d'offre n'existe pas ou si la référence n'est pas valide ou si aucune référence n'est fournie.

## retireOfferTemplates

```
public WSGenerateOfferCodeResponse generateOfferCodes  
(String userCredential, String partitionName, Locale requestedLocale,  
String offerName, WSReference template)  
throws CampaignServicesException
```

Retire un ou plusieurs modèles d'offres spécifiés.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*wsCampaignReference* : référence de la campagne parent.

### Renvoie

Renvoie l'objet de type WSRetireOfferTemplatesResponse.

### Erreurs

Emet l'exception CampaignServicesException si le modèle d'offre n'existe pas ou si la référence n'est pas valide ou si aucune référence n'est fournie.

## getOffersAndListsByPage

```
public WSGetOffersAndListsByPageResponse getOffersAndListsByPage  
(String userCredential, String partitionName, Locale requestedLocale,  
WSComponentTypeEnum type,  
int pageSize, int pageOffset)  
throws CampaignServicesException
```

Répertorie les offres ou les listes d'offres par page.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*type* : type indiquant si des offres ou listes d'offres sont demandées.

*pageSize* : nombre maximal de composants correspondants à renvoyer pour la page.

*pageOffset* : décalage de début de tous les composants possibles pour commencer l'énumération. Par exemple, si l'énumération correspond à 1000 offres et si cette

valeur est définie sur 10, la page commencera au 11ème composant. Une `RangeException` est émise si le décalage fourni est hors plage.

## Renvoie

Renvoie l'objet de type `WSGetOffersAndListsByPageResponse`.

## Erreurs

Aucune.

## bulkCreateOffers

```
WSOfferInfoStatus[] bulkCreateOffers(String authorizationLoginName,  
    String partitionName, Locale requestedLocale,  
    String securityPolicyName, String templateName, long folderID,  
    WSBulkOfferInfo[] offers)  
    throws CampaignServicesException;
```

Crée des offres en masse à l'aide des attributs pour chaque offre spécifiée dans le paramètre *offers*. Toutes les offres sont créées sous l'ID de dossier (*folderID*) spécifié à l'aide du nom de modèle (*templateName*) spécifié.

### Paramètre

*authorizationLoginName* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*securityPolicyName* : nom facultatif de la stratégie de sécurité de campagne à utiliser pour créer l'offre. Si ce paramètre n'est pas défini, la stratégie globale est utilisée.

*templateName* : nom du modèle d'offre existant dans le système. Toutes les offres sont créées à l'aide de ce modèle.

*folderID* : ID du dossier Offre où sont créées les offres. Cet ID est validé et une exception est émise si l'ID n'est pas valide.

*offers* : matrice d'objets `WSBulkOfferInfo` définissant le nom et les attributs de l'offre. Reportez-vous au type de données `WSBulkOfferInfo` pour plus de détails.

## Renvoie

une matrice d'instances `WSOfferInfoStatus` pour chaque offre. Contient le statut et les informations d'offre. Le statut indique si la création de l'offre a abouti ou non.

## Erreurs

`CampaignServicesException`

## getOfferListDetails

```
public WSGetOfferListDetailsResponse getOfferListDetails(String userCredential,  
    String partitionName, Locale requestedLocale, WSReference listReference)  
    throws CampaignServicesException {
```

Répertorie les détails de la liste d'offres spécifiée.

## Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*listReference* : référence à la liste d'offres.

## Renvoie

Renvoie l'objet de type `WSGetOfferListDetailsResponse`.

## Erreurs

Emet l'exception `CampaignServicesException` si les références de liste ne sont pas valides.

## getOfferListMembers

```
public WSGetOfferListMembersResponse getOfferListMembers  
(String userCredential, String partitionName, Locale requestedLocale,  
 WSReference listReference)  
 throws CampaignServicesException {
```

Répertorie les informations d'offres dans la liste d'offres spécifiée.

## Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*listReference* : référence à la liste d'offres.

## Renvoie

Renvoie l'objet de type `WSDeleteOffersAndListsResponse`.

## Erreurs

Emet l'exception `CampaignServicesException` si l'ID de liste d'offres n'est pas valide.

## getOffersByQuery

```
public WSGetOffersByQueryResponse getOffersByQuery(String user_credential,  
 String partition_name, Locale locale, String query, Integer maxSize,  
 Boolean includeSubFolder, WSReference[] scopeFolders)  
 throws CampaignServicesException
```

Répertorie les offres correspondant aux offres fournies.



## Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*query* : requête de recherche d'offres. Le format de la requête est le même que celui utilisé dans la liste d'offres dynamiques.

*maxSize* : nombre maximal d'enregistrements à répertorier.

*includeSubFolder* : indicateur qui indique si un sous-dossier doit être inclus dans la recherche.

*scopeFolders* : liste des références de dossier à rechercher pour les offres.

## Renvoie

Renvoie l'objet de type `WSGetOffersByQueryResponse`.

## Erreurs

Emet l'exception `CampaignServicesException` si les références de dossier ne sont pas valides.

## retireOfferLists

```
public void retireOfferLists(String user_credential, String partition_name,  
Locale locale, WSReference[] wsReferences)  
    throws CampaignServicesException
```

Retire une ou plusieurs listes d'offres spécifiées.

## Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*wsReferences* : référence aux listes d'offres.

## Renvoie

Aucune.

## Erreurs

Emet l'exception `CampaignServicesException` si les références de liste ne sont pas valides.

## createFolder

```
public WSCreateFolderResponse createFolder(String userCredential,  
String partitionName, Locale requestedLocale,String name,  
String description,String securityPolicyName,  
long parentFolderId,WSFolderTypeEnum folderType,  
WSApplicationTypeEnum createdBy,long creatorObjectId)  
throws CampaignServicesException
```

Crée un dossier de type campaign/offer/sessions/segments.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*name* : nom du dossier à créer.

*description* : description du nouveau dossier.

*securityPolicyName* : nom de la stratégie de sécurité à utiliser.

*parentFolderId* : (facultatif) id du dossier parent. S'il n'est pas fourni, le dossier est créé au niveau root.

*folderType* : type de folder-Offer/session/campaign/segment.

*createdBy* : (facultatif) l'objet de type WSApplicationTypeEnum indique l'application qui a créé le dossier. Les valeurs possibles sont Campaign, Plan, Collaborate et eMessage. S'il n'est pas fourni, Campaign est utilisé.

*creatorObjectId* : (facultatif) utilisé par le plan pour lier le dossier dans la campagne au dossier dans le plan.

### Renvoie

Renvoie l'objet de type WSCreateFolderResponse.

### Erreurs

Emet l'exception CampaignServicesException si le type de dossier n'est pas valide ou en double.

## editFolder

```
public WSEditFolderResponse editFolder(String userCredential,  
String partitionName, Locale requestedLocale, long folderId,  
String name,String description, WSFolderTypeEnum folderType,  
Long creatorObjectId, boolean clearExisting)  
throws CampaignServicesException
```

Met à jour le dossier indiqué.

## Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*id* : ID du dossier à mettre à jour.

*name* : nom du dossier à créer.

*description* : description du nouveau dossier.

*folderType* : type de folder-Offer/session/campaign/segment.

*creatorObjectId* : (facultatif) utilisé par le plan pour lier le dossier dans la campagne au dossier dans le plan.

*clearExisting* : non utilisé actuellement.

## Renvoie

Renvoie l'objet de type `WSEditFolderResponse`.

## Erreurs

Emet l'exception `CampaignServicesException` si le type de dossier n'est pas valide ou en double ou si l'ID de dossier n'est pas valide.

## getSubFoldersList

```
public WSGetSubFolderListResponse getSubFoldersList(String user_credential,  
String partition_name, Locale locale, WSReference parentFolder,  
WSFolderTypeEnum folderType)  
    throws CampaignServicesException
```

Répertorie tous les sous-dossiers dans le dossier spécifié.

## Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*parentFolder* : référence du dossier pour lequel tous les sous-dossiers sont demandés.

*folderType* : type du dossier.

## Renvoie

Renvoie l'objet de type `WSGetSubFolderListResponse`.

## Erreurs

Emet l'exception CampaignServicesException si la référence de dossier n'est pas valide.

## moveFolders

```
public WSMoveFolderResponse moveFolders(String userCredential,  
String partitionName, Locale requestedLocale, Long[] folderId,  
long parentFolder, long destinationFolder,  
WSFolderTypeEnum folderType)  
throws CampaignServicesException
```

Déplace le dossier spécifié vers un autre dossier parent.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*folderId* : ID du dossier à déplacer.

*parentFolder* : ID du dossier parent.

*destinationFolder* : ID du dossier de destination vers lequel le dossier indiqué est déplacé.

*folderType* : type de folder-Offer/session/campaign/segment.

### Renvoie

Renvoie l'objet de type WSMoveFolderResponse.

### Erreurs

Emet l'exception CampaignServicesException si le type de dossier ou l'ID de dossier parent n'est pas valide ou si l'ID de dossier n'est pas valide.

## deleteFolders

```
public WSDeleteFolderResponse deleteFolders(String userCredential,  
String partitionName, Locale requestedLocale, Long[] folderId,  
long parentFolder, boolean deleteChilds,  
WSFolderTypeEnum folderType)  
throws CampaignServicesException
```

Supprime les dossiers spécifiés ainsi que tous les éléments qu'ils contiennent.

### Paramètres

*userCredential* : données d'identification de l'utilisateur client.

*partitionName* : nom facultatif de la partition de campagne à utiliser.

*requestedLocale* : paramètres régionaux facultatifs à utiliser pour cette demande.

*folderId* : ID du dossier à supprimer.

*parentFolder* : ID du dossier parent.

*deleteChilds* : l'indicateur indique si toutes les dépendances du dossier doivent être supprimées. S'il a la valeur *false*, aucune dépendance n'est supprimée.

*folderType* : type de folder-Offer/session/campaign/segment.

## **Renvois**

Renvoie l'objet de type *WSDeleteFolderResponse*.

## **Erreurs**

Emet l'exception *CampaignServicesException* si le type de dossier n'est pas valide ou si l'ID de dossier n'est pas valide.



---

## Chapitre 5. Utilisation de l'API

Cette section explique comment utiliser l'API Campaign Web Services. Elle contient également un exemple d'utilisation de l'API Campaign Service pour créer une offre dans Campaign.

Deux approches permettent d'utiliser l'API Campaign Services :

- «Utilisation du fichier .jar de l'API client»
- «Utilisation du fichier WSDL», à la page 59

---

### Utilisation du fichier .jar de l'API client

IBM Campaign offre une API client qui utilise des services Web SOAP pour interagir avec l'application Web Campaign. Cet encapsuleur est intégré dans un fichier .jar que l'application client peut utiliser pour appeler l'API Campaign.

Ce fichier .jar est disponible sous :

```
<CAMPAIGN_HOME>/devkits/CampaignServicesAPI/lib/  
CampaignServicesClient30.jar
```

L'exemple suivant illustre la création d'une nouvelle offre au niveau du dossier Offre racine dans Campaign. Le même exemple est disponible sous :

```
<CAMPAIGN_HOME>/devkits/CampaignServicesAPI/samples/OfferAPI.java
```

**Remarque :** L'exemple utilise certaines valeurs factices pour les paramètres ; vos valeurs réelles pourront être différentes.

De plus, l'URL des services Web Campaign est la suivante : `http://host:port/Campaign/services/CampaignServices30Service`, où `host` et `port` correspondent au nom d'hôte et au numéro de port de la machine sur laquelle l'application Web Campaign est déployée.

Si vous utilisez un exemple fourni, veillez à le modifier pour l'adapter à votre environnement client.

#### OfferAPI.java

```
import java.net.URL;  
import java.util.Locale;  
import com.unica.publicapi.campaign.campaignservices.CampaignServicesException;  
import com.unica.publicapi.campaign.campaignservices.attribute.metadata.  
    IAttribute Metadata;  
import com.unica.publicapi.campaign.campaignservices.soap.v30.  
    CampaignServices30SoapClient;  
import com.unica.publicapi.campaign.campaignservices.soap.v30.WSAttribute;  
import com.unica.publicapi.campaign.campaignservices.soap.v30.WSOfferInfo;  
import com.unica.publicapi.campaign.campaignservices.utils.WSAttributeUtils;  
  
/**  
 * This is the sample java client class that shows the usage of Campaign's SOAP  
 services API.  
 * This sample uses CampaignServices30SoapClient facade to interact with Campaign  
 web service. */
```

```

* Here the creation of Offer is shown. Please refer to the API guide for
more details.
*
* @author AGijare
*
*/
public class OfferAPI {

    /**
     * @param args
     */
    protected static CampaignServices30SoapClient CLIENT = null;

    private static void setup(){
        try {
            String protocol = "http"; //http or https
            String host = "localhost"; //Host name of deployed Campaign.
            Use proper host name here.
            int port = 7001; //port number of deployed Campaign
            long timeOut = 2*60*1000; // 2 minutes
            String servicesURI = "/Campaign/services/CampaignServices30Service";
            CLIENT = new CampaignServices30SoapClient(
                new URL(protocol, host, port, servicesURI),
                timeOut);
        } catch (Exception exception) {
            exception.printStackTrace();
            System.exit(-1);
        }
    }

    public static void main(String[] args) {
        //Please change the values of following variables to match your
environment.
        String userName = "user_name"; //login user name
        String partitionName = "partition1"; //Use proper partition name of
Campaign
        Locale loc = Locale.US;
        String securityPolicy = "Global"; //Use your security policy of
Campaign

        String offerName = "Offer1";
        String offerTemplate = "Offer Template"; // Template from which
Offer will be created.
        long folderID = 1002; //Actual ID of the folder where this offer
will be created. For folderID <=0, offer will be created at root level.
        //Attributes of Offer
        WSAttribute[] wsAttributes = {
            WSAttributeUtils.getWSTextAttribute(IAAttributeMeta
data.AC_OFFER_DESCRIPTION_ATTRIBUTE_NAME, null, new String[]{"description "
+ System.currentTimeMillis()})
        };

        setup();

        try {
            WSOfferInfo wsOfferInfo = CLIENT.createOffer(userName,
partitionName, loc, securityPolicy,
offerName, folderID, offerTemplate, wsAttributes);
            System.out.println("Created offer: " + wsOfferInfo.getName());
        } catch (CampaignServicesException e) {
            e.printStackTrace();
        }
    }
}

```

**Remarque :** Pour compiler et exécuter l'exemple Java présenté ci-dessus, vous devez inclure tous les fichiers .jar dépendants dans le chemin d'accès aux classes



Java. Le fichier CampaignServicesClient30.jar est dépendant des fichiers .jar du moteur SOAP Apache AXIS2 et d'autres fichiers .jar Apache courants, disponibles dans le fichier Campaign.war qui se trouve dans <CAMPAIGN\_HOME>/Campaign.war. Vous devez extraire les fichiers .jar de Campaign.war, puis les inclure dans le chemin d'accès aux classes Java.

---

## Utilisation du fichier WSDL

L'API Campaign Services peut également être appelée à l'aide du fichier WSDL Campaign Web Services :

CampaignServices30.wsdl

disponible sous :

<http://host:port/Campaign/services/CampaignServices30Service?wsdl>

ou dans la distribution Campaign sous :

<CAMPAIGN\_HOME>/devkits/CampaignServicesAPI/lib/

L'application Java client doit utiliser les classes et les modules de remplacement générés à partir du fichier WSDL à l'aide d'un outil de conversion WSDL-Java tiers. IBM recommande l'utilisation d'Apache AXIS.

Les Javadocs créés à partir des modules de remplacement et des classes générés à partir du fichier WSDL à l'aide d'Apache AXIS2 sont disponibles sous :

<CAMPAIGN\_HOME>/devkits/CampaignServicesAPI/javadocs/index.html

**Remarque :** Tous les fichiers .jar dépendants doivent être inclus dans le chemin d'accès aux classes Java. Le fichier CampaignServicesClient30.jar est dépendant des fichiers .jar du moteur SOAP Apache AXIS2 et d'autres fichiers .jar Apache courants, disponibles dans le fichier Campaign.war qui se trouve dans <CAMPAIGN\_HOME>/Campaign.war. Vous devez extraire les fichiers .jar de Campaign.war, puis les inclure dans le chemin d'accès aux classes Java.

---

## Considérations relatives aux performances

Le profil de performances de mise en oeuvre de l'API CampaignServices actuel est similaire à celui de l'application tel qu'il est utilisé via l'interface graphique. Certaines API sont conçues explicitement pour les performances. En particulier, l'API `listCampaignsByPage()` permet une pagination relativement efficace.

De par sa nature, l'interface SOAP introduit un temps d'attente et un temps système car toutes les données sont converties au format XML qui, dans certains cas, est plutôt prolix. Par exemple, un simple appel SOAP de bouclage peut prendre 100 ms sur un réseau classique (Java 1.4.x était encore plus lent). L'API est optimisée pour des cas d'utilisation métier classiques du portail et d'autres applications client, voir `listOffersByPage()`, de sorte que les performances SOAP soient appropriées.

Toutefois, le client doit veiller à ne pas surcharger le service CampaignServices normal fourni aux demandes d'utilisateur Web. En général, les besoins de traitement d'un utilisateur d'API ne dépassent pas ceux d'un utilisateur Web IBM Campaign classique.

---

## Packaging

Cette spécification est fournie dans le cadre des kits de développement logiciels (devkits) CampaignServices installés avec IBM Campaign.

Le répertoire devkits établi par le programme d'installation inclut des exemples, des scripts de génération et de texte, du Javadoc pour les classes et interfaces publiques, des notes sur l'édition.

---

## Avant de contacter le support technique IBM

Si vous rencontrez un problème que vous ne parvenez pas à résoudre en consultant la documentation, le correspondant désigné pour le support technique de votre entreprise peut contacter le support technique d'IBM. Utilisez ces instructions pour garantir la résolution efficace de votre problème.

Si vous n'êtes pas le correspondant désigné pour le support technique dans votre société, contactez l'administrateur IBM pour plus d'informations.

### Informations à réunir

Avant de contacter le support technique d'IBM, rassemblez les informations suivantes :

- une brève description de la nature du problème,
- Messages d'erreur détaillés s'affichant lorsque le problème se produit
- La liste des étapes complètes permettant de reproduire l'erreur.
- Fichiers journaux, fichiers de session, fichiers de configuration et fichiers de données connexes
- Informations relatives à votre environnement (produit et système ), que vous pouvez obtenir en suivant la procédure décrite sous "Informations sur le système".

### Informations système

Lorsque vous appellerez le support technique d'IBM, vous devrez sans doute fournir des informations relatives à votre environnement.

Si le problème rencontré ne vous empêche pas de vous connecter, vous trouverez la plupart de ces informations sur la page A propos de, qui fournit des informations sur les applications IBM installées.

Pour accéder à la page A propos de, sélectionnez **Aide > A propos de**. Si vous ne pouvez pas accéder à la page A propos de, recherchez le fichier `version.txt` qui se trouve sous le répertoire d'installation de votre application.

### Informations de contact du support technique d'IBM

Pour savoir comment contacter le support technique d'IBM, consultez le site Web de support technique IBM : ([http://www.ibm.com/support/entry/portal/open\\_service\\_request](http://www.ibm.com/support/entry/portal/open_service_request)).

**Remarque :** Pour entrer une demande de support, vous devez vous connecter avec un compte IBM. Ce compte doit être associé à votre numéro client IBM. Pour en savoir plus sur l'association de votre compte avec votre numéro client IBM, voir **Ressources de support > Soutien de logiciels autorisés** sur le portail de support.



---

## Remarques

Le présent document peut contenir des informations ou des références concernant certains produits, logiciels ou services IBM non annoncés dans ce pays. Pour plus de détails, référez-vous aux documents d'annonce disponibles dans votre pays, ou adressez-vous à votre partenaire commercial IBM. Toute référence à un produit, programme ou service IBM n'implique pas que seul ce produit, programme ou service IBM puisse être utilisé. Tout autre élément fonctionnellement équivalent peut être utilisé, s'il n'enfreint aucun droit d'IBM. Il est de la responsabilité de l'utilisateur d'évaluer et de vérifier lui-même les installations et applications réalisées avec des produits, logiciels ou services non expressément référencés par IBM.

IBM peut détenir des brevets ou des demandes de brevet couvrant les produits mentionnés dans le présent document. La remise de ce document ne vous accorde aucun droit de licence sur ces brevets ou demandes de brevet. Si vous désirez recevoir des informations concernant l'acquisition de licences, veuillez en faire la demande par écrit à l'adresse suivante :

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

Pour le Canada, veuillez adresser votre courrier à :

IBM Director of Commercial Relations  
IBM Canada Ltd  
3600 Steeles Avenue East  
Markham, Ontario  
L3R 9Z7 Canada

Les informations sur les licences concernant les produits utilisant un jeu de caractères double octet peuvent être obtenues auprès d'IBM Intellectual Property Department dans votre pays ou par écrit à l'adresse :

Intellectual Property Licensing  
Legal and Intellectual Property Law  
IBM Japan, Ltd.  
19-21, Nihonbashi-Hakozakicho, Chuo-ku  
Tokyo 103-8510, Japan

Le paragraphe suivant ne s'applique ni au Royaume-Uni, ni dans aucun pays dans lequel il serait contraire aux lois locales. LE PRESENT DOCUMENT EST LIVRE EN L'ETAT SANS AUCUNE GARANTIE EXPLICITE OU IMPLICITE. IBM DECLINE NOTAMMENT TOUTE RESPONSABILITE RELATIVE A CES INFORMATIONS EN CAS DE CONTREFACON AINSI QU'EN CAS DE DEFAUT D'APTITUDE A L'EXECUTION D'UN TRAVAIL DONNE. Certaines juridictions n'autorisent pas l'exclusion des garanties implicites, auquel cas l'exclusion ci-dessus ne vous sera pas applicable.

Le présent document peut contenir des inexactitudes ou des coquilles. Ce document est mis à jour périodiquement. Chaque nouvelle édition inclut les mises à jour. IBM peut, à tout moment et sans préavis, modifier les produits et logiciels décrits dans ce document.

Les références à des sites Web non IBM sont fournies à titre d'information uniquement et n'impliquent en aucun cas une adhésion aux données qu'ils contiennent. Les éléments figurant sur ces sites Web ne font pas partie des éléments du présent produit IBM et l'utilisation de ces sites relève de votre seule responsabilité.

IBM pourra utiliser ou diffuser, de toute manière qu'elle jugera appropriée et sans aucune obligation de sa part, tout ou partie des informations qui lui seront fournies.

Les licenciés souhaitant obtenir des informations permettant : (i) l'échange des données entre des logiciels créés de façon indépendante et d'autres logiciels (dont celui-ci), et (ii) l'utilisation mutuelle des données ainsi échangées, doivent adresser leur demande à :

IBM Corporation  
170 Tracer Lane  
Waltham, MA 02451  
U.S.A.

Ces informations peuvent être soumises à des conditions particulières, prévoyant notamment le paiement d'une redevance.

Le logiciel sous licence décrit dans le présent document et tous les éléments sous licence disponibles s'y rapportant sont fournis par IBM conformément aux dispositions de l'ICA (IBM Customer Agreement), des conditions internationales d'utilisation des logiciels IBM ou de tout autre accord équivalent.

Les données de performance indiquées dans ce document ont été déterminées dans un environnement contrôlé. Par conséquent, les résultats peuvent varier de manière significative selon l'environnement d'exploitation utilisé. Certaines mesures évaluées sur des systèmes en cours de développement ne sont pas garanties sur tous les systèmes disponibles. En outre, elles peuvent résulter d'extrapolations. Les résultats peuvent donc varier. Il incombe aux utilisateurs de ce document de vérifier si ces données sont applicables à leur environnement d'exploitation.

Les informations concernant des produits non IBM ont été obtenues auprès des fournisseurs de ces produits, par l'intermédiaire d'annonces publiques ou via d'autres sources disponibles. IBM n'a pas testé ces produits et ne peut confirmer l'exactitude de leurs performances ni leur compatibilité. Elle ne peut recevoir aucune réclamation concernant des produits non IBM. Toute question concernant les performances de produits non IBM doit être adressée aux fournisseurs de ces produits.

Toute instruction relative aux intentions d'IBM pour ses opérations à venir est susceptible d'être modifiée ou annulée sans préavis, et doit être considérée uniquement comme un objectif.

Tous les tarifs indiqués sont les prix de vente actuels suggérés par IBM et sont susceptibles d'être changés sans préavis. Les tarifs appliqués peuvent varier selon les revendeurs.

Le présent document peut contenir des exemples de données et de rapports utilisés couramment dans l'environnement professionnel. Ces exemples mentionnent des noms fictifs d'individus, de sociétés, de marques ou de produits à des fins illustratives ou explicatives uniquement. Toute ressemblance avec des noms d'individus, de sociétés ou des données réelles serait purement fortuite.

#### LICENCE DE COPYRIGHT :

Le présent guide contient des exemples de programmes d'application en langage source destinés à illustrer les techniques de programmation sur différentes plateformes d'exploitation. Vous avez le droit de copier, de modifier et de distribuer ces exemples de programmes sous quelque forme que ce soit et sans paiement d'aucune redevance à IBM à des fins de développement, d'utilisation, de vente ou de distribution de programmes d'application conformes à l'interface de programme d'application de la plateforme pour lesquels ils ont été écrits. Ces exemples de programmes n'ont pas été rigoureusement testés dans toutes les conditions. Par conséquent, IBM ne peut garantir expressément ou implicitement la fiabilité, la maintenabilité ou le fonctionnement de ces programmes. Les exemples de programmes sont fournis en l'état, sans garantie d'aucune sorte. IBM ne pourra en aucun cas être tenue responsable des dommages liés à l'utilisation des exemples de programmes.

Si vous visualisez ces informations en ligne, il se peut que les photographies et illustrations en couleur n'apparaissent pas à l'écran.

---

## Marques

IBM, le logo IBM et [ibm.com](http://ibm.com) sont des marques d'International Business Machines Corp. dans de nombreux pays. Les autres noms de produits et de services peuvent être des marques d'IBM ou d'autres sociétés. La liste actualisée de toutes les marques d'IBM est disponible sur la page Web «Copyright and trademark information» à l'adresse [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

---

## Règles de confidentialité et conditions d'utilisation

Les produits IBM Software, notamment les logiciels sous forme de services (“Offres logicielles”), peuvent utiliser des cookies ou d'autres technologies pour collecter des informations d'utilisation en vue d'améliorer l'expérience de l'utilisateur final, d'ajuster les interactions avec l'utilisateur final ou à d'autres fins. Un cookie est une donnée qu'un site Web peut envoyer à votre navigateur et qui peut ensuite être stockée sur votre ordinateur sous la forme d'une balise identifiant ce dernier. Dans la plupart des cas, aucune information personnelle n'est collectée par ces cookies. Si vous utilisez une offre logicielle qui vous permet de collecter des informations personnelles via des cookies et des technologies similaires, tenez compte des spécificités suivantes.

En fonction de la configuration déployée, cette offre logicielle peut utiliser des cookies de session et des cookies permanents qui collectent le nom d'utilisateur de chaque utilisateur ainsi que d'autres informations personnelles à des fins de gestion des sessions, de convivialité améliorée pour l'utilisateur ou d'autres objectifs de suivi de l'utilisation ou fonctionnels. Ces cookies peuvent être désactivés mais leur désactivation élimine également la fonctionnalité qu'ils activent.

Diverses juridictions régulent la collecte d'informations personnelles via les cookies et autres technologies similaires. Si la configuration déployée pour cette offre

logicielle vous permet, en tant que client, de collecter des informations personnelles d'utilisateurs finaux via des cookies et autres technologies, vous devez rechercher les conseils juridiques sur les lois applicables à ces collectes de données, notamment toutes les exigences relatives à l'indication de consignes et de consentements, le cas échéant.

IBM exige que les clients (1) fournissent un lien clair et visible vers les conditions d'utilisation du site Web du client (par exemple les règles de confidentialité) qui comprenne un lien vers les collectes de données et les pratiques d'utilisation d'IBM et du client, (2) indiquent que des cookies et des gifs et pixels invisibles sont placés sur l'ordinateur du visiteur par IBM pour le compte du client avec une explication de l'objectif de cette technologie et (3), selon les conditions requises par la loi, obtiennent le consentement des visiteurs du site Web avant de placer les cookies et les gifs et pixels invisibles par le client ou IBM sur leurs unités.

Pour plus d'informations sur l'utilisation des diverses technologies, notamment des cookies, reportez-vous à la section intitulée "Cookies, Web Beacons and Other Technologies" d'IBM's Online Privacy Statement, à l'adresse : <http://www.ibm.com/privacy/details/us/en>.





