Package ‘rdfp’

June 6, 2019

**Title**  An Implementation of the 'DoubleClick for Publishers' API

**Version**  0.1.4

**Date**  2019-06-05

**Description**  Functions to interact with the 'Google DoubleClick for Publishers (DFP)' API &lt;https://developers.google.com/ad-manager/api/start&gt; (recently renamed to 'Google Ad Manager'). This package is automatically compiled from the API WSDL (Web Service Description Language) files to dictate how the API is structured. Theoretically, all API actions are possible using this package; however, care must be taken to format the inputs correctly and parse the outputs correctly. Please see the 'Google Ad Manager' API reference &lt;https://developers.google.com/ad-manager/api/rel_notes&gt; and this package's website &lt;https://stevenmmortimer.github.io/rdfp/&gt; for more information, documentation, and examples.

**URL**  https://github.com/StevenMMortimer/rdfp

**BugReports**  https://github.com/StevenMMortimer/rdfp/issues

**Encoding**  UTF-8

**Depends**  R (&gt;= 3.5.0)

**License**  MIT + file LICENSE

**LazyData**  true

**Imports**  utils (&gt;= 3.5.0), methods (&gt;= 3.5.0), httr (&gt;= 1.4.0), curl (&gt;= 3.3), plyr (&gt;= 1.8.4), XML (&gt;= 3.98-1.19), dplyr (&gt;= 0.8.0), xml2 (&gt;= 1.2.0), readr (&gt;= 1.3.1), data.table (&gt;= 1.12.0), purrr, lubridate

**Suggests**  knitr, testthat, rmarkdown, here

**RoxygenNote**  6.1.1

**Collate**  'ActivityGroupService.R' 'ActivityService.R'
'AdExclusionRuleService.R' 'AdRuleService.R'
'AdjustmentService.R' 'AudienceSegmentService.R'
'BaseRateService.R' 'CdnConfigurationService.R'
'CmsMetadataService.R' 'CompanyService.R' 'ContactService.R'
'ContentBundleService.R' 'ContentService.R' 'CreativeService.R'
R topics documented:

'CreativeSetService.R' 'CreativeTemplateService.R'
'CreativeWrapperService.R' 'CustomFieldService.R'
'CustomTargetingService.R' 'DaiAuthenticationKeyService.R'
'ExchangeRateService.R' 'ForecastService.R'
'InventoryService.R' 'LabelService.R'
'LineItemCreativeAssociationService.R' 'LineItemService.R'
'LineItemTemplateService.R' 'LiveStreamEventService.R'
'MobileApplicationService.R' 'NativeStyleService.R'
'NetworkService.R' 'OrderService.R' 'PackageService.R'
'PlacementService.R' 'PremiumRateService.R'
'ProductPackageItemService.R' 'ProductPackageService.R'
'ProductService.R' 'ProposalService.R'
'ProposalLineItemService.R' 'ProposalService.R'
'PublisherQueryLanguageService.R' 'RateCardService.R'
'ReconciliationLineItemReportService.R'
'ReconciliationOrderReportService.R'
'ReconciliationReportRowService.R'
'ReconciliationReportService.R' 'ReportService.R'
'SuggestedAdUnitService.R' 'TargetingPresetService.R'
'TeamService.R' 'UserService.R' 'UserTeamAssociationService.R'
'WorkflowRequestService.R' 'dfp_auth.R'
'dfp_options_settings.R' 'dfp_service_endpoints.R'
'dfp_utils.R' 'dfp_utils_xml.R' 'rdfp.R'

VignetteBuilder knitr

NeedsCompilation no

Author Steven M. Mortimer [aut, cre],
    Jennifer Bryan [ctb],
    Joanna Zhao [ctb]

Maintainer Steven M. Mortimer <reportmort@gmail.com>

Repository CRAN

Date/Publication 2019-06-05 22:00:19 UTC

R topics documented:

dfp_auth ................................. 7
dfp_createActivities .................. 8
dfp_createActivityGroups ............. 9
dfp_createAdExclusionRules .......... 10
dfp_createAdRules .................... 11
dfp_createAdUnits .................... 12
dfp_createAudienceSegments .......... 13
dfp_createBaseRates .................. 14
dfp_createCdnConfigurations .......... 15
dfp_createCompanies ................ 16
dfp_createContacts .................. 17
dfp_createContentBundles ............ 18
topics documented:

- dfp_createCreatives ........................................... 19
- dfp_createCreativeSet ......................................... 20
- dfp_createCreativeWrappers .................................. 21
- dfp_createCustomFieldOptions ............................... 22
- dfp_createCustomFields ...................................... 23
- dfp_createCustomTargetingKeys ......................... 24
- dfp_createCustomTargetingValues ..................... 25
- dfp_createDaiAuthenticationKeys ....................... 26
- dfp_createExchangeRates .................................. 27
- dfp_createLabels ............................................. 28
- dfp_createLineItemCreativeAssociations .................. 29
- dfp_createLineItems ......................................... 30
- dfp_createLiveStreamEvents ................................ 31
- dfp_createMobileApplications ............................ 32
- dfp_createNativeStyles ..................................... 33
- dfp_createOrders ............................................. 34
- dfp_createPackages .......................................... 35
- dfp_createPlacements ........................................ 36
- dfp_createPremiumRates .................................... 37
- dfp_createProductPackageItems .................. 38
- dfp_createProductPackages ............................... 39
- dfp_createProductTemplates ......................... 40
- dfp_createProposalLineItems ............................. 41
- dfp_createProposals .......................................... 42
- dfp_createRateCards ....................................... 43
- dfp_createTeams ............................................. 44
- dfp_createUsers ............................................. 45
- dfp_createUserTeamAssociations ....................... 46
- dfp_date_to_list ............................................. 47
- dfp_full_report_wrapper .................................. 47
- dfp_getActivitiesByStatement ............................. 48
- dfp_getActivityGroupsByStatement ...................... 50
- dfp_getAdExclusionRulesByStatement .................... 51
- dfp_getAdRulesByStatement ................................ 52
- dfp_getAdSpotsByStatement ................................ 53
- dfp_getAdUnitsByStatement ................................ 53
- dfp_getAdUnitSizesByStatement ....................... 55
- dfp_getAllNetworks .......................................... 56
- dfp_getAllRoles ............................................... 57
- dfp_getAudienceSegmentsByStatement .................... 57
- dfp_getAvailabilityForecast .............................. 59
- dfp_getAvailabilityForecastById ...................... 60
- dfp_getBaseRatesByStatement ............................. 61
- dfp_getCdnConfigurationsByStatement .................. 62
- dfp_getCmsMetadataKeysByStatement ..................... 63
- dfp_getCmsMetadataValuesByStatement .................. 64
- dfp_getCompaniesByStatement ............................ 65
- dfp_getContactsByStatement .............................. 66
dfp_getContentBundlesByStatement ........................................... 67
dfp_getContentByStatement .................................................. 68
dfp_getContentByStatementAndCustomTargetingValue .................. 69
dfp_getCreativesByStatement ............................................... 70
dfp_getCreativeSetsByStatement ......................................... 71
dfp_getCreativeTemplatesByStatement ................................... 72
dfp_getCreativeWrappersByStatement ................................. 73
dfp_getCurrentNetwork ..................................................... 74
dfp_getCurrentUser ........................................................... 74
dfp_getCustomFieldOption ................................................. 75
dfp_getCustomFieldsByStatement ...................................... 76
dfp_getCustomTargetingKeysByStatement ............................ 77
dfp_getCustomTargetingValuesByStatement ......................... 78
dfp_getDaiAuthenticationKeysByStatement ............................ 79
dfp_getDeliveryForecast ...................................................... 80
dfp_getDeliveryForecastByIds ............................................. 81
dfp_getExchangeRatesByStatement ..................................... 82
dfp_getLabelsByStatement .................................................. 83
dfp_getLineItemCreativeAssociationsByStatement .................. 84
dfp_getLineItemsByStatement ............................................. 85
dfp_getLineItemTemplatesByStatement ................................ 86
dfp_getLiveStreamEventsByStatement ................................ 87
dfp_getMarketplaceCommentsByStatement ............................ 88
dfp_getMobileApplicationsByStatement .............................. 89
dfp_getNativeStylesByStatement ....................................... 90
dfp_getOrdersByStatement ................................................. 91
dfp_getPackagesByStatement .............................................. 92
dfp_getPlacementsByStatement ......................................... 93
dfp_getPremiumRatesByStatement ..................................... 94
dfp_getPreviewUrl .......................................................... 95
dfp_getProductPackageItemsByStatement ............................. 96
dfp_getProductPackagesByStatement ................................ 97
dfp_getProductsByStatement ............................................. 98
dfp_getProductTemplatesByStatement ................................ 99
dfp_getProposalLineItemsByStatement ................................ 100
dfp_getProposalsByStatement ........................................... 101
dfp_getRateCardsByStatement .......................................... 102
dfp_getReconciliationLineItemReportsByStatement ................. 104
dfp_getReconciliationOrderReportsByStatement .................... 105
dfp_getReconciliationReportRowsByStatement ....................... 106
dfp_getReconciliationReportsByStatement ......................... 107
dfp_getReportDownloadURL ............................................... 108
dfp_getReportDownloadUrlWithOptions .............................. 110
dfp_getReportJobStatus ................................................... 111
dfp_getSavedQueriesByStatement ....................................... 112
dfp_getSuggestedAdUnitsByStatement ................................ 113
dfp_getTargetingPresetSByStatement .................................. 114
dfp_getTeamsByStatement ........................................... 115
dfp_getTrafficAdjustmentsByStatement ........................... 116
dfp_getUsersByStatement ........................................... 117
dfp_getUserTeamAssociationsByStatement ......................... 118
dfp_getWorkflowRequestsByStatement ............................. 119
dfp_hasCustomPacingCurve ......................................... 120
dfp_makeTestNetwork .............................................. 121
dfp_performAdExclusionRuleAction ................................ 122
dfp_performAdRuleAction ........................................... 123
dfp_performAdUnitAction ........................................... 124
dfp_performAudienceSegmentAction ................................. 124
dfp_performBaseRateAction ......................................... 125
dfp_performCdnConfigurationAction ............................... 126
dfp_performContentBundleAction .................................. 127
dfp_performCreativeWrapperAction ................................ 127
dfp_performCustomFieldAction .................................... 128
dfp_performCustomTargetingKeyAction ............................ 129
dfp_performCustomTargetingValueAction ......................... 130
dfp_performDaiAuthenticationKeyAction ......................... 130
dfp_performExchangeRateAction ................................... 131
dfp_performLabelAction ............................................ 132
dfp_performLineItemAction ........................................ 133
dfp_performLineItemCreativeAssociationAction ................ 134
dfp_performLiveStreamEventAction ................................ 134
dfp_performMobileApplicationAction ............................... 135
dfp_performNativeStyleAction ..................................... 136
dfp_performOrderAction ............................................ 137
dfp_performPackageAction ......................................... 137
dfp_performPlacementAction ....................................... 138
dfp_performProductAction ......................................... 139
dfp_performProductPackageAction ................................. 140
dfp_performProductPackageItemAction ........................... 140
dfp_performProductTemplateAction ................................ 141
dfp_performProposalAction ........................................ 142
dfp_performProposalLineItemAction ............................... 143
dfp_performRateCardAction ....................................... 144
dfp_performReconciliationOrderReportAction ................... 144
dfp_performSuggestedAdUnitAction ................................. 145
dfp_performTeamAction ............................................ 146
dfp_performUserAction ............................................. 147
dfp_performUserTeamAssociationAction ........................... 148
dfp_performWorkflowRequestAction ............................... 148
dfp_registerSessionsForMonitoring ............................... 149
dfp_report_url_to_dataframe ...................................... 150
dfp_runReportJob .................................................... 151
dfp_select ............................................................ 152
dfp_updateActivities ................................................ 153
dfp_updateActivityGroups ......................................... 154
dfp_updateAdExclusionRules ........................................ 155
dfp_updateAdRules ...................................................... 156
dfp_updateAdUnits ....................................................... 156
dfp_updateAudienceSegments ......................................... 157
dfp_updateBaseRates .................................................... 158
dfp_updateCdnConfigurations ......................................... 159
dfp_updateCompanies .................................................. 159
dfp_updateContacts ..................................................... 160
dfp_updateContentBundles ............................................. 161
dfp_updateCreatives .................................................... 162
dfp_updateCreativeSet .................................................. 162
dfp_updateCreativeWrappers ......................................... 163
dfp_updateCustomFieldOptions ...................................... 164
dfp_updateCustomFields ................................................. 165
dfp_updateCustomTargetingKeys .................................... 165
dfp_updateCustomTargetingValues .................................. 166
dfp_updateDaiAuthenticationKeys ................................... 167
dfp_updateExchangeRates .............................................. 168
dfp_updateLabels ......................................................... 168
dfp_updateLineItemCreativeAssociations ............................ 169
dfp_updateLineItems ..................................................... 170
dfp_updateLiveStreamEvents ......................................... 171
dfp_updateMobileApplications ....................................... 171
dfp_updateNativeStyles ............................................... 172
dfp_updateNetwork ...................................................... 173
dfp_updateOrders ......................................................... 174
dfp_updatePackages ..................................................... 174
dfp_updatePlacements ................................................... 175
dfp_updatePremiumRates .............................................. 176
dfp_updateProductPackageItems ..................................... 177
dfp_updateProductPackages .......................................... 177
dfp_updateProducts ...................................................... 178
dfp_updateProductTemplates ......................................... 179
dfp_updateProposalLineItems ......................................... 180
dfp_updateProposals ..................................................... 180
dfp_updateRateCards .................................................... 181
dfp_updateReconciliationLineItemReports ......................... 182
dfp_updateReconciliationOrderReports ............................. 183
dfp_updateReconciliationReportRows ............................... 183
dfp_updateReconciliationReports ................................... 184
dfp_updateTeams ........................................................ 185
dfp_updateTrafficAdjustments ....................................... 186
dfp_updateUsers ........................................................ 187
dfp_updateUserTeamAssociations .................................... 187
rdfp ................................................................. 188

Index ................................................................. 189
dfp_auth

Description

Authorize rdfp to access your Google user data. You will be directed to a web browser, asked to sign in to your Google account, and to grant rdfp access to user data for Double Click for Publishers. These user credentials are cached in a file named .httr-oauth in the current working directory, from where they can be automatically refreshed, as necessary.

Usage

```r
dfp_auth(token = NULL, new_user = FALSE,
    addtl_scopes = c("https://spreadsheets.google.com/feeds",
        "https://www.googleapis.com/auth/drive",
        "https://www.googleapis.com/auth/spreadsheets",
        "https://www.googleapis.com/auth/presentations",
        "https://www.googleapis.com/auth/analytics",
        "https://www.googleapis.com/auth/yt-analytics.readonly",
        "https://www.googleapis.com/auth/gmail.readonly",
        "https://www.googleapis.com/auth/gmail.compose",
        "https://www.googleapis.com/auth/gmail.send"),
    key = getOption("rdfp.client_id"),
    secret = getOption("rdfp.client_secret"),
    cache = getOption("rdfp.httr_oauth_cache"), verbose = TRUE)
```

Arguments

- **token**: an actual token object or the path to a valid token stored as an .rds file
- **new_user**: logical, defaults to FALSE. Set to TRUE if you want to wipe the slate clean and re-authenticate with the same or different Google account. This deletes the .httr-oauth-rdfp file in current working directory.
- **addtl_scopes**: character, strings that indicate additional Google services the client should authorize. Use this when trying to generate a token that will work to authenticate to other packages using Google services, such as the googlesheets package or RGoogleAnalytics package.
- **key**, **secret**: the "Client ID" and "Client secret" for the application
- **cache**: logical indicating if rdfp should cache credentials in the default cache file .httr-oauth-rdfp
- **verbose**: a logical indicating if messages should be printed

Details

Most users, most of the time, do not need to call this function explicitly – it will be triggered by the first action that requires authorization. Even when called, the default arguments will often suffice. However, when necessary, this function allows the user to
• store a token – the token is invisibly returned and can be assigned to an object or written to an .rds file
• read the token from an .rds file or pre-existing object in the workspace
• provide your own app key and secret – this requires setting up a new project in Google Developers Console
• prevent caching of credentials in .httr-oauth-rdfp

In a call to dfp_auth, the user can provide the token, app key and secret explicitly and can dictate whether credentials will be cached in .httr-oauth-rdfp. They must be specified.

To set options in a more persistent way, predefine one or more of them with lines like this in a .Rprofile file:

```r
options(rdfp.network_code = "12345678",
    rdfp.application_name = "MyApp",
    rdfp.client_id = "012345678901-99thisisatest99.apps.googleusercontent.com",
    rdfp.client_secret = "Th1s1sMyClient5cr3t",
    rdfp.httr_oauth_cache = FALSE)
```

See Startup for possible locations for this file and the implications thereof.

More detail is available from Using OAuth 2.0 to Access Google APIs.

Value

an OAuth token object, specifically a `Token2.0`, invisibly

---

**dfp_createActivities**

**ActivityService**

**Description**

Provides methods for creating, updating and retrieving Activity objects.

**Usage**

```r
dfp_createActivities(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML
Details

An activity group contains Activity objects. Activities have a many-to-one relationship with activity groups, meaning each activity can belong to only one activity group, but activity groups can have multiple activities. An activity group can be used to manage the activities it contains.

createActivities

Creates a new Activity objects.

Value

a data.frame or list containing all the elements of a createActivitiesResponse

See Also

Google Documentation for createActivities

Examples

```r
## Not run:
res <- dfp_createActivities(request_data)

## End(Not run)
```

---

**dfp_createActivityGroups**

*ActivityGroupService*

Description

Provides methods for creating, updating and retrieving ActivityGroup objects. An activity group contains Activity objects. Activities have a many-to-one relationship with activity groups, meaning each activity can belong to only one activity group, but activity groups can have multiple activities. An activity group can be used to manage the activities it contains.

Usage

```r
dfp_createActivityGroups(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- **request_data** a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df** a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose** a boolean indicating whether to print the service URL and POSTed XML
dfp_createAdExclusionRules

Details

createActivityGroups
Creates a new ActivityGroup objects.

Value

da.data.frame or list containing all the elements of a createActivityGroupsResponse

See Also

Google Documentation for createActivityGroups

Examples

## Not run:
res <- dfp_createActivityGroups(request_data)

## End(Not run)

---

dfp_createAdExclusionRules

AdExclusionRuleService

Description

Provides methods for creating, updating and retrieving AdExclusionRule objects. An AdExclusionRule provides a way to block specified ads from showing on portions of your site. Each rule specifies the inventory on which the rule is in effect, and the labels to block on that inventory.

Usage

dfp_createAdExclusionRules(request_data, as_df = TRUE, verbose = FALSE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_data</td>
<td>a list or data.frame of data elements to be formatted for a SOAP request</td>
</tr>
<tr>
<td></td>
<td>(XML format, but passed as character string)</td>
</tr>
<tr>
<td>as_df</td>
<td>a boolean indicating whether to attempt to parse the result into a data.frame</td>
</tr>
<tr>
<td>verbose</td>
<td>a boolean indicating whether to print the service URL and POSTed XML</td>
</tr>
</tbody>
</table>

Details

createAdExclusionRules
Creates new AdExclusionRule objects.
**Value**

a data.frame or list containing all the elements of a `createAdExclusionRulesResponse`

**See Also**

Google Documentation for `createAdExclusionRules`

**Examples**

```r
## Not run:
res <- dfp_createAdExclusionRules(request_data)
## End(Not run)
```

**dfp_createAdRules**

`AdRuleService`

**Description**

Provides methods for creating, updating and retrieving AdRule objects.

**Usage**

```r
dfp_createAdRules(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Details**

Ad rules contain data that the ad server uses to generate a playlist of video ads.

createAdRules

Creates new AdRule objects.

**Value**

a data.frame or list containing all the elements of a `createAdRulesResponse`

**See Also**

Google Documentation for `createAdRules`
**dfp_createAdUnits**

**Examples**

```r
## Not run:
res <- dfp_createAdRules(request_data)
```

```r
## End(Not run)
```

---

**InventoryService**

**Description**

Provides operations for creating, updating and retrieving AdUnit objects.

**Usage**

```r
dfp_createAdUnits(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Details**

Line items connect a creative with its associated ad unit through targeting. An ad unit represents a piece of inventory within a publisher. It contains all the settings that need to be associated with the inventory in order to serve ads. For example, the ad unit contains creative size restrictions and AdSense settings.

createAdUnits

Creates new AdUnit objects.

**Value**

A data.frame or list containing all the elements of a createAdUnitsResponse

**See Also**

[Google Documentation for createAdUnits](#)

**Examples**

```r
## Not run:
res <- dfp_createAdUnits(request_data)
```

```r
## End(Not run)
```
**dfp_createAudienceSegments**

*AudienceSegmentService*

**Description**

Provides operations for creating, updating and retrieving AudienceSegment objects.

**Usage**

```
dfp_createAudienceSegments(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Details**

createAudienceSegments

Creates new RuleBasedFirstPartyAudienceSegment objects.

**Value**

a data.frame or list containing all the elements of a createAudienceSegmentsResponse

**See Also**

*Google Documentation for createAudienceSegments*

**Examples**

```r
## Not run:
res <- dfp_createAudienceSegments(request_data)

## End(Not run)
```
dfp_createBaseRates

BaseRateService

Description

Provides methods for managing BaseRate objects. To use this service, you need to have the new sales management solution enabled on your network. If you do not see a "Sales" tab in <a href="https://www.google.com/dfp">DoubleClick for Publishers (DFP)</a>, you will not be able to use this service.

Usage

dfp_createBaseRates(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Details

createBaseRates

Creates a list of new BaseRate objects.

Value

a data.frame or list containing all the elements of a createBaseRatesResponse

See Also

Google Documentation for createBaseRates

Examples

## Not run:
res <- dfp_createBaseRates(request_data)

## End(Not run)
**Description**

Provides methods for creating, updating and retrieving CdnConfiguration objects.

**Usage**

```r
dfp_createCdnConfigurations(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- **request_data**
  - a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**
  - a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**
  - a boolean indicating whether to print the service URL and POSTed XML

**Details**

- **createCdnConfigurations**
  - Creates new CdnConfiguration objects.

**Value**

- a data.frame or list containing all the elements of a createCdnConfigurationsResponse

**See Also**

- [Google Documentation for createCdnConfigurations](#)

**Examples**

```r
## Not run:
res <- dfp_createCdnConfigurations(request_data)
## End(Not run)
```
dfp_createCompanies  *CompanyService*

**Description**

Provides operations for creating, updating and retrieving Company objects.

**Usage**

```
dfp_createCompanies(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Details**

- `createCompanies`
  - Creates new Company objects.

**Value**

A data.frame or list containing all the elements of a `createCompaniesResponse`

**See Also**

[Google Documentation for `createCompanies`]

**Examples**

```r
## Not run:
request_data <- list(companies=list(name="TestCompany1",
    type='HOUSE_ADVERTISER',
    address='123 Main St Hometown, FL USA',
    email='testcompany1@gmail.com',
    comment='API Test'))

result <- dfp_createCompanies(request_data)
## End(Not run)
```
**dfp_createContacts**  

**ContactService**

**Description**

Provides methods for creating, updating and retrieving Contact objects.

**Usage**

```r
dfp_createContacts(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data` a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Details**

`createContacts`

Creates new Contact objects.

**Value**

a `data.frame` or list containing all the elements of a `createContactsResponse`

**See Also**

[Google Documentation for createContacts](https://developers.google.com/dfp/docs/reference/rest/v1.3)

**Examples**

```r
## Not run:
request_data <- list(contact=list(name="TestContact1",  
companyID=dfp_createCompanies_result$id,  
status='UNINVITED',  
cellPhone='(888) 999-7777',  
comment='API Test',  
email='testcontact1@gmail.com'))

result <- dfp_createContacts(request_data)

## End(Not run)
```
dfp_createContentBundles

Description

Provides methods for creating, updating and retrieving ContentBundle objects. A ContentBundle is a grouping of Content that match filter rules as well as taking into account explicitly included or excluded Content.

Usage

```
dfp_createContentBundles(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Details

createContentBundles

Creates new ContentBundle objects.

Value

A `data.frame` or list containing all the elements of a `createContentBundlesResponse`

See Also

Google Documentation for `createContentBundles`

Examples

```r
## Not run:
res <- dfp_createContentBundles(request_data)
## End(Not run)
```
dfp_createCreatives  CreativeService

Description

Provides methods for adding, updating and retrieving Creative objects.

Usage

dfp_createCreatives(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df  a boolean indicating whether to attempt to parse the result into a data.frame

verbose  a boolean indicating whether to print the service URL and POSTed XML

Details

For a creative to run, it must be associated with a LineItem managed by the LineItemCreativeAssociationService.#' Read more about creatives on the <a href="https://support.google.com/dfp_premium/answer/3185155">DFP Help Center</a>.

createCreatives

Creates new Creative objects.

Value

a data.frame or list containing all the elements of a createCreativesResponse

See Also

Google Documentation for createCreatives

Examples

```r
## Not run:
res <- dfp_createCreatives(request_data)

## End(Not run)
```
dfp_createCreativeSet  CreativeSetService

Description

Provides methods for adding, updating and retrieving CreativeSet objects.

Usage

dfp_createCreativeSet(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Details

createCreativeSet

Creates a new CreativeSet.

Value

A data.frame or list containing all the elements of a createCreativeSetResponse

See Also

Google Documentation for createCreativeSet

Examples

```r
## Not run:
res <- dfp_createCreativeSet(request_data)

## End(Not run)
```
dfp_createCreativeWrappers

CreativeWrapperService

Description

Provides methods for the creation and management of creative wrappers. CreativeWrapper Cre-
avativeWrappers allow HTML snippets to be served along with creatives. Creative wrappers must be
associated with a LabelType#CREATIVE_WRAPPER label and applied to ad units by AdUnit#appliedLabels.

Usage

dfp_createCreativeWrappers(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request
             (XML format, but passed as character string)
as_df        a boolean indicating whether to attempt to parse the result into a data.frame
verbose      a boolean indicating whether to print the service URL and POSTed XML

Details

createCreativeWrappers

Creates a new CreativeWrapper objects. The following fields are required:

- CreativeWrapper labelId
- CreativeWrapper ordering
- CreativeWrapper header or CreativeWrapper footer

Value

a data.frame or list containing all the elements of a createCreativeWrappersResponse

See Also

Google Documentation for createCreativeWrappers

Examples

```r
## Not run:
res <- dfp_createCreativeWrappers(request_data)

## End(Not run)
```
dfp_createCustomFieldOptions

CustomFieldService

Description

Provides methods for the creation and management of CustomField objects.

Usage

dfp_createCustomFieldOptions(request_data, as_df = TRUE, 
verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request 
(XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Details

createCustomFieldOptions

Creates new CustomFieldOption objects. The following fields are required:

• CustomFieldOption displayName
• CustomFieldOption customFieldId

Value

a data.frame or list containing all the elements of a createCustomFieldOptionsResponse

See Also

Google Documentation for createCustomFieldOptions

Examples

## Not run:
request_data <- data.frame(customFieldId=rep(dfp_createCustomFields_result$id, 3), 
displayName=c('Morning', 'Afternoon', 'Evening'))
result <- dfp_createCustomFieldOptions(request_data)

## End(Not run)
dfp_createCustomFields

createCustomFields

Description

Creates new CustomField objects. The following fields are required:

- CustomField name
- CustomField entityType
- CustomField dataType
- CustomField visibility

Usage

dfp_createCustomFields(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df a boolean indicating whether to attempt to parse the result into a data.frame

verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a createCustomFieldsResponse

See Also

Google Documentation for createCustomFields

Examples

## Not run:
request_data <- data.frame(name='Timing',
    description='The time that this creative runs.',
    entityType='CREATIVE',
    dataType='DROP_DOWN',
    visibility='FULL')
result <- dfp_createCustomFields(request_data)

## End(Not run)
Description

Provides operations for creating, updating and retrieving CustomTargetingKey and CustomTargetingValue objects.

Usage

```
dfp_createCustomTargetingKeys(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- **request_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML

Details

- **createCustomTargetingKeys**
  Creates new CustomTargetingKey objects. The following fields are required:
  - CustomTargetingKey name
  - CustomTargetingKey type

Value

A data.frame or list containing all the elements of a createCustomTargetingKeysResponse

See Also

- Google Documentation for createCustomTargetingKeys

Examples

```
## Not run:
request_data <- list(keys=list(name="Test1",
                           displayName="TestKey1",
                           type='FREEFORM'))
result <- dfp_createCustomTargetingKeys(request_data)

## End(Not run)
```
dfp_createCustomTargetingValues

dfpxcreatecustomtargetingvalues

createCustomTargetingValues

Description

Creates new CustomTargetingValue objects. The following fields are required:

- CustomTargetingValue customTargetingKeyId
- CustomTargetingValue name

Usage

```
dfp_createCustomTargetingValues(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a createCustomTargetingValuesResponse

See Also

Google Documentation for createCustomTargetingValues

Examples

```r
## Not run:
request_data <- data.frame(customTargetingKeyId=rep(created_targeting_key$id,2),
                          name=c('TestValue','TestValue2'),
                          displayName=c('TestValue','TestValue2'),
                          matchType=rep('EXACT', 2))
result <- dfp_createCustomTargetingValues(request_data)

## End(Not run)
```
dfp_createDaiAuthenticationKeys

DaiAuthenticationKeyService

Description

Provides methods for creating, updating and retrieving DaiAuthenticationKey objects.

Usage

dfp_createDaiAuthenticationKeys(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df         a boolean indicating whether to attempt to parse the result into a data.frame
verbose       a boolean indicating whether to print the service URL and POSTed XML

Details

createDaiAuthenticationKeys

Creates new DaiAuthenticationKey objects. The following fields are required:

• DaiAuthenticationKey name

Value

a data.frame or list containing all the elements of a createDaiAuthenticationKeysResponse

See Also

Google Documentation for createDaiAuthenticationKeys

Examples

## Not run:
res <- dfp_createDaiAuthenticationKeys(request_data)

## End(Not run)
dfp_createExchangeRates

ExchangeRateService

Description

Provides methods for adding, updating and retrieving ExchangeRate objects.

Usage

dfp_createExchangeRates(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  

a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df  

a boolean indicating whether to attempt to parse the result into a data.frame

verbose  

a boolean indicating whether to print the service URL and POSTed XML

Details

createExchangeRates

Creates new ExchangeRate objects. For each exchange rate, the following fields are required:

- ExchangeRate currencyCode
- ExchangeRate exchangeRate when ExchangeRate refreshRate is ExchangeRateRefreshRate FIXED

Value

a data.frame or list containing all the elements of a createExchangeRatesResponse

See Also

Google Documentation for createExchangeRates

Examples

```r
## Not run:
res <- dfp_createExchangeRates(request_data)

## End(Not run)
```
**Description**

Provides methods for the creation and management of Labels.

**Usage**

```r
dfp_createLabels(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Details**

`createLabels`

Creates new Label objects.

**Value**

A `data.frame` or list containing all the elements of a createLabelsResponse

**See Also**

Google Documentation for `createLabels`

**Examples**

```r
## Not run:
request_data <- data.frame(name="Auto - Competitive Exclusion",
                          description=paste0("A label to prevent two different car ",
                                       "companies from showing ads together"),
                          types='COMPETITIVE_EXCLUSION')
request_data <- list('labels'=hypothetical_label)
result <- dfp_createLabels(request_data)

## End(Not run)
```
dfp_createLineItemCreativeAssociations

LineItemCreativeAssociationService

Description

Provides operations for creating, updating and retrieving LineItemCreativeAssociation objects. A line item creative association (LICA) associates a Creative with a LineItem. When a line item is selected to serve, the LICAs specify which creatives can appear for the ad units that are targeted by the line item. In order to be associated with a line item, the creative must have a size that exists within the attribute LineItem#creativeSizes." Each LICA has a start and end date and time that defines when the creative should be displayed." To read more about associating creatives with line items, see this <a href="https://support.google.com/dfp_premium/answer/3187916">DFP Help Center</a> article.

Usage

dfp_createLineItemCreativeAssociations(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Details

createLineItemCreativeAssociations
Creates new LineItemCreativeAssociation objects

Value

a data.frame or list containing all the elements of a createLineItemCreativeAssociationsResponse

See Also

Google Documentation for createLineItemCreativeAssociations

Examples

## Not run:
res <- dfp_createLineItemCreativeAssociations(request_data)

## End(Not run)
dfp_createLineItems  

Description

#' Provides methods for creating, updating and retrieving LineItem objects.

Usage

dfp_createLineItems(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df  a boolean indicating whether to attempt to parse the result into a data.frame

verbose  a boolean indicating whether to print the service URL and POSTed XML

Details

Line items define the campaign. For example, line items define:

- a budget
- a span of time to run
- ad unit targeting

In short, line items connect all of the elements of an ad campaign. Line items and creatives can be associated with each other through LineItemCreativeAssociation objects. An ad unit will host a creative through both this association and the LineItem#targeting to it.

createLineItems

Creates new LineItem objects.

Value

a data.frame or list containing all the elements of a createLineItemsResponse

See Also

Google Documentation for createLineItems

Examples

## Not run:
res <- dfp_createLineItems(request_data)

## End(Not run)
dfp_createLiveStreamEvents

**LiveStreamEventService**

**Description**

Provides methods for creating, updating and retrieving LiveStreamEvent objects. This feature is not yet openly available for DFP Video publishers. Publishers will need to apply for access for this feature through their account managers.

**Usage**

```r
dfp_createLiveStreamEvents(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- **request_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML

**Details**

**createLiveStreamEvents**

Creates new LiveStreamEvent objects. The following fields are required:
- LiveStreamEvent name
- LiveStreamEvent startDateTime
- LiveStreamEvent endDateTime
- LiveStreamEvent contentUrls
- LiveStreamEvent adTags

**Value**

a data.frame or list containing all the elements of a createLiveStreamEventsResponse

**See Also**

Google Documentation for createLiveStreamEvents

**Examples**

```r
## Not run:
res <- dfp_createLiveStreamEvents(request_data)

## End(Not run)
```
dfp_createMobileApplications

MobileApplicationService

Description

Provides methods for retrieving MobileApplication objects.

Usage

```
dfp_createMobileApplications(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Details

createMobileApplications

Creates and claims MobileApplication mobile applications to be used for targeting in the network.

Value

A data.frame or list containing all the elements of a createMobileApplicationsResponse

See Also

Google Documentation for createMobileApplications

Examples

```r
## Not run:
res <- dfp_createMobileApplications(request_data)
## End(Not run)
```
dfp_createNativeStyles

NativeStyleService

Description

Provides methods for creating and retrieving NativeStyle objects.

Usage

dfp_createNativeStyles(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df  a boolean indicating whether to attempt to parse the result into a data.frame
verbose  a boolean indicating whether to print the service URL and POSTed XML

Details

createNativeStyles

Creates new NativeStyle objects.

Value

a data.frame or list containing all the elements of a createNativeStylesResponse

See Also

Google Documentation for createNativeStyles

Examples

## Not run:
res <- dfp_createNativeStyles(request_data)

## End(Not run)
**dfp_createOrders**

**OrderService**

**Description**

Provides methods for creating, updating and retrieving Order objects.

**Usage**

```r
dfp_createOrders(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data` a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Details**

An order is a grouping of LineItem objects. Line items have a many-to-one relationship with orders, meaning each line item can belong to only one order, but orders can have multiple line items. An order can be used to manage the line items it contains.

- `createOrders`
  - Creates new Order objects.

**Value**

- a data.frame or list containing all the elements of a createOrdersResponse

**See Also**

Google Documentation for createOrders

**Examples**

```r
## Not run:
request_data <- list('filterStatement'=list('query'="WHERE name = 'TestCompany'"))
dfp_getCompaniesByStatement_result <- dfp_getCompaniesByStatement(request_data)

request_data <- list(list(name='TestOrder',
                         startDateTime=list(date=list(year=2018, month=12, day=1),
                                            hour=0,
                                            minute=0,
                                            second=0,
                                            timeZoneID='America/New_York'),
                         endDateTime=list(date=list(year=2018, month=12, day=31)),
```
dfp_createPackages

Description

Provides methods for creating, updating and retrieving Package objects.

Usage

dfp_createPackages(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df  a boolean indicating whether to attempt to parse the result into a data.frame

verbose  a boolean indicating whether to print the service URL and POSTed XML

Details

To use this service, you need to have the new sales management solution enabled on your network. If you do not see a "Sales" tab in <a href="https://www.google.com/dfp">DoubleClick for Publishers (DFP)</a>, you will not be able to use this service.

createPackages

Creates new Package objects. For each package, the following fields are required:

• Package proposalId
• Package productPackageId
• Package name

Value

a data.frame or list containing all the elements of a createPackagesResponse
See Also

Google Documentation for createPackages

Examples

```r
## Not run:
res <- dfp_createPackages(request_data)
```

```r
## End(Not run)
```

---

**dfp_createPlacements**  
*PlacementService*

**Description**

Provides methods for creating, updating and retrieving Placement objects.

**Usage**

`dfp_createPlacements(request_data, as_df = TRUE, verbose = FALSE)`

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Details**

You can use a placement to group ad units. For example, you might have a placement that focuses on sports sites, which may be spread across different branches of your inventory. You might also have a "fire sale" placement that includes ad units that have not been selling and are consequently priced very attractively.

createPlacements

Creates new Placement objects.

**Value**

A data.frame or list containing all the elements of a createPlacementsResponse

**See Also**

Google Documentation for createPlacements
dfp_createPremiumRates

**Examples**

```r
## Not run:
res <- dfp_createPlacements(request_data)

## End(Not run)
```

---

**dfp_createPremiumRates**  
*PremiumRateService*

**Description**

Provides methods for managing PremiumRate objects. To use this service, you need to have the new sales management solution enabled on your network. If you do not see a "Sales" tab in [DoubleClick for Publishers (DFP)](https://www.google.com/dfp), you will not be able to use this service.

**Usage**

```
dfp_createPremiumRates(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`  
  a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)

- `as_df`  
  a boolean indicating whether to attempt to parse the result into a `data.frame`

- `verbose`  
  a boolean indicating whether to print the service URL and POSTed XML

**Details**

- `createPremiumRates`
  Creates a list of new PremiumRate objects.

**Value**

- a `data.frame` or list containing all the elements of a `createPremiumRatesResponse`

**See Also**

[Google Documentation for createPremiumRates](https://www.google.com/dfp)

**Examples**

```r
## Not run:
res <- dfp_createPremiumRates(request_data)

## End(Not run)
```
dfp_createProductPackageItems

*ProductPackageItemService*

**Description**

Provides methods for creating and retrieving ProductPackageItem objects.

**Usage**

`dfp_createProductPackageItems(request_data, as_df = TRUE, verbose = FALSE)`

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Details**

A ProductPackageItem represents a product which will be associated with a ProductPackage. To use this service, you need to have the new sales management solution enabled on your network. If you do not see a "Sales" tab in [DoubleClick for Publishers (DFP)](https://www.google.com/dfp), you will not be able to use this service.

`createProductPackageItems`

Creates new ProductPackageItem objects.

**Value**

a data.frame or list containing all the elements of a `createProductPackageItemsResponse`

**See Also**


**Examples**

```r
# Not run:
res <- dfp_createProductPackageItems(request_data)

# End(Not run)
```
**dfp_createProductPackages**

*ProductPackageService*

**Description**

Provides methods for updating and retrieving ProductPackage objects. A ProductPackage represents a group of products which will be sold together.

**Usage**

```
dfp_createProductPackages(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Details**

To use this service, you need to have the new sales management solution enabled on your network. If you do not see a "Sales" tab in <a href="https://www.google.com/dfp">DoubleClick for Publishers (DFP)</a>, you will not be able to use this service.

`createProductPackages`

Creates new ProductPackage objects.

**Value**

- a `data.frame` or `list` containing all the elements of a `createProductPackagesResponse`

**See Also**

Google Documentation for `createProductPackages`

**Examples**

```r
## Not run:
res <- dfp_createProductPackages(request_data)

## End(Not run)```
dfp_createProductTemplates

ProductTemplateService

Description

Provides methods for creating, updating and retrieving ProductTemplate objects. A product template is used to generate a set of products. Products allow inventory managers to control what salespeople can sell.

Usage

dfp_createProductTemplates(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df         a boolean indicating whether to attempt to parse the result into a data.frame
verbose       a boolean indicating whether to print the service URL and POSTed XML

Details

To use this service, you need to have the new sales management solution enabled on your network. If you do not see a "Sales" tab in &lt;a href="https://www.google.com/dfp">DoubleClick for Publishers (DFP)&lt;/a&gt;, you will not be able to use this service.

createProductTemplates

Creates new ProductTemplate objects.

Value

a data.frame or list containing all the elements of a createProductTemplatesResponse

See Also

Google Documentation for createProductTemplates

Examples

## Not run:
res <- dfp_createProductTemplates(request_data)

## End(Not run)
dfp_createProposalLineItems

ProposalLineItemService

Description

Provides methods for creating, updating and retrieving ProposalLineItem objects. To use this service, you need to have the new sales management solution enabled on your network. If you do not see a "Sales" tab in <a href="https://www.google.com/dfp">DoubleClick for Publishers (DFP)</a>, you will not be able to use this service.

Usage

dfp_createProposalLineItems(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df a boolean indicating whether to attempt to parse the result into a data.frame

verbose a boolean indicating whether to print the service URL and POSTed XML

Details

createProposalLineItems

Creates new ProposalLineItem objects.

Value

a data.frame or list containing all the elements of a createProposalLineItemsResponse

See Also

Google Documentation for createProposalLineItems

Examples

## Not run:
res <- dfp_createProposalLineItems(request_data)

## End(Not run)
**dfp_createProposals** *ProposalService*

**Description**

Provides methods for adding, updating and retrieving Proposal objects.

**Usage**

```
dfp_createProposals(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data` : a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` : a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose` : a boolean indicating whether to print the service URL and POSTed XML

**Details**

- `createProposals`
  Creates new Proposal objects. For each proposal, the following fields are required:

  - Proposal name

**Value**

- a `data.frame` or `list` containing all the elements of a `createProposalsResponse`

**See Also**

- Google Documentation for `createProposals`

**Examples**

```r
## Not run:
res <- dfp_createProposals(request_data)

## End(Not run)
```
Description

Provides methods for managing RateCard objects. To use this service, you need to have the new sales management solution enabled on your network. If you do not see a "Sales" tab in <a href="https://www.google.com/dfp">DoubleClick for Publishers (DFP)</a>, you will not be able to use this service.

Usage

dfp_createRateCards(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- **request_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML

Details

createRateCards

Creates a list of RateCard objects. Rate cards must be activated before being associated with proposal line items and products.

Value

a data.frame or list containing all the elements of a createRateCardsResponse

See Also

Google Documentation for createRateCards

Examples

```r
## Not run:
res <- dfp_createRateCards(request_data)

## End(Not run)
```
dfp_createTeams  TeamService

Description

Provides methods for creating, updating, and retrieving Team objects.

Usage

dfp_createTeams(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df  a boolean indicating whether to attempt to parse the result into a data.frame
verbose  a boolean indicating whether to print the service URL and POSTed XML

Details

Teams are used to group users in order to define access to entities such as companies, inventory and orders.
createTeams

Creates new Team objects. The following fields are required:

- Team name

Value

a data.frame or list containing all the elements of a createTeamsResponse

See Also

Google Documentation for createTeams

Examples

```r
## Not run:
request_data <- list(teams=list(name="TestTeam1",
description='API Test Team 1',
hasAllCompanies='true',
hasAllInventory='true',
teamAccessType='READ_WRITE'))
result <- dfp_createTeams(request_data)
## End(Not run)
```
dfp_createUsers

UserService

Description

Provides operations for creating, updating and retrieving User objects.

Usage

dfp_createUsers(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data    a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df           a boolean indicating whether to attempt to parse the result into a data.frame
verbose         a boolean indicating whether to print the service URL and POSTed XML

Details

A user is assigned one of several different roles. Each Role type has a unique ID that is used to identify that role in an organization. Role types and their IDs can be retrieved by invoking #getAllRoles.

createUsers

Creates new User objects.

Value

a data.frame or list containing all the elements of a createUsersResponse

See Also

Google Documentation for createUsers

Examples

```r
## Not run:
request_data <- data.frame(name = paste0("TestUser", 1:3),
    email = paste0('testuser', 1:3, '@gmail.com'),
    roleid = rep(-1, 3))
result <- dfp_createUsers(request_data)
```

## End(Not run)
**dfp_createUserTeamAssociations**

*UserTeamAssociationService*

**Description**

Provides methods for creating, updating, and retrieving UserTeamAssociation objects. UserTeamAssociation objects are used to add users to teams in order to define access to entities such as companies, inventory and orders and to override the team’s access type to orders for a user.

**Usage**

```
dfp_createUserTeamAssociations(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Details**

`createUserTeamAssociations`

Creates new UserTeamAssociation objects.

**Value**

A data.frame or list containing all the elements of a createUserTeamAssociationsResponse

**See Also**

Google Documentation for createUserTeamAssociations

**Examples**

```r
## Not run:
request_data <- data.frame(teamid=rep(dfp_createTeams_result$id, 3),
                           userid=rep(dfp_createUsers_result$id, 3))
result <- dfp_createUserTeamAssociations(request_data)
## End(Not run)
```
**dfp_date_to_list**

Format a datetime for DFP

**Description**

Take a datetime representation in R and convert it to the list required by DFP to indicate a datetime.

**Usage**

```r
dfp_date_to_list(this_date, daytime = c("beginning", "end"),
                 timeZoneId = Sys.timezone(), ensure_today_works = TRUE)
```

**Arguments**

- `this_date` Date; formatted as Date, POSIXct, or POSIXlt
- `daytime` character; either "beginning" or "end" so that the function knows which hours to set if needed
- `timeZoneId` character; a string indicating the timezone that should be used. The timezone ID must be in Time_Zone database
- `ensure_today_works` logical; an indicator that will automatically offset the current time by 1 hour so that forecasts will actually work. If you try to forecast from a timestamp of now, then by the time you submit it to the ForecastService it will already be too late to be in the future.

**Value**

a list formatted to the spec required for StartDateTime or EndDateTime

**Examples**

```r
dfp_date_to_list(Sys.Date()+1)
```

---

**dfp_full_report_wrapper**

Take report request and return data.frame

**Description**

Take a report request and manage all aspects for user until returning a data.frame or error.

**Usage**

```r
dfp_full_report_wrapper(request_data",
                      check_interval=3,
                      max_tries=20,
                      verbose=FALSE)
```
**Arguments**

- **request_data**
  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

- **check_interval**
  a numeric specifying seconds to wait between report status requests to check if complete

- **max_tries**
  a numeric specifying the maximum number of times to check whether the report is complete before the function essentially times out

- **verbose**
  a logical indicating whether to print the report URL

**Value**

a data.frame of report results as specified by the request_data

**See Also**

- dfp_runReportJob
- dfp_getReportJobStatus
- dfp_getReportDownloadURL

**Examples**

```r
## Not run:
request_data <- list(reportJob =
  list(reportQuery =
    list(dimensions = 'MONTH_AND_YEAR',
      dimensions = 'AD_UNIT_ID',
      dimensions = 'AD_UNIT_NAME',
      dimensions = 'ADVERTISER_NAME',
      dimensions = 'ORDER_NAME',
      dimensions = 'LINE_ITEM_NAME',
      adUnitView = 'FLAT',
      columns = 'AD_SERVER_IMPRESSIONS',
      columns = 'AD_SERVER_CLICKS',
      dateRangeType = 'LAST_WEEK'))

report_data <- dfp_full_report_wrapper(request_data)

## End(Not run)
```
### Description

Gets an ActivityPage of Activity objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- name
- expectedURL
- status
- activityGroupId

### Usage

```r
dfp_getActivitiesByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

### Arguments

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

### Value

A `data.frame` or list containing all the elements of a `getActivitiesByStatementResponse`

### See Also

- Google Documentation for `getActivitiesByStatement`

### Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getActivitiesByStatement(dat)
## End(Not run)
```
dfp_getActivityGroupsByStatement

getActivityGroupsByStatement

Description

Gets an ActivityGroupPage of ActivityGroup objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- name
- impressionsLookback
- clicksLookback
- status

Usage

dfp_getActivityGroupsByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getActivityGroupsByStatementResponse

See Also

Google Documentation for getActivityGroupsByStatement

Examples

## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getActivityGroupsByStatement(dat)

## End(Not run)
Description

Gets a AdExclusionRulePage of AdExclusionRule objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- name
- status

Usage

dfp_getAdExclusionRulesByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df a boolean indicating whether to attempt to parse the result into a data.frame

verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getAdExclusionRulesByStatementResponse

See Also

Google Documentation for getAdExclusionRulesByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getAdExclusionRulesByStatement(dat)
## End(Not run)
```
dfp_getAdRulesByStatement

getAdRulesByStatement

Description

Gets an AdRulePage of AdRule objects that satisfy the given (@link Statement query). The following fields are supported for filtering:

- id
- name
- priority
- status

Usage

dfp_getAdRulesByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

da data.frame or list containing all the elements of a getAdRulesByStatementResponse

See Also

Google Documentation for getAdRulesByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getAdRulesByStatement(dat)

## End(Not run)
```
**dfp_getAdSpotsByStatement**

### Description

Gets a AdSpotPage of AdSpot objects that satisfy the given [@link Statement query].

### Usage

```r
dfp_getAdSpotsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

### Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

### Value

A data.frame or list containing all the elements of a getAdSpotsByStatementResponse

### See Also

[Google Documentation for getAdSpotsByStatement](#)

### Examples

```r
# Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getAdSpotsByStatement(dat)

# End(Not run)
```

---

**dfp_getAdUnitsByStatement**

### getAdUnitsByStatement

---
**dfp_getAdUnitsByStatement**

**Description**

Gets a AdUnitPage of AdUnit objects that satisfy the given Statement query. The following fields are supported for filtering:

- adUnitCode
- id
- name
- parentId
- status
- lastModifiedDateTime

**Usage**

`dfp_getAdUnitsByStatement(request_data, as_df = FALSE, verbose = FALSE)`

**Arguments**

- `request_data` a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a getAdUnitsByStatementResponse

**See Also**

[Google Documentation for getAdUnitsByStatement](#)

**Examples**

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getAdUnitsByStatement(dat)
## End(Not run)
```
dfp_getAdUnitSizesByStatement

Description

Gets a set of AdUnitSize objects that satisfy the given Statement query. The following fields are supported for filtering:

- targetPlatform

Usage

dfp_getAdUnitSizesByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

- a data.frame or list containing all the elements of a getAdUnitSizesByStatementResponse

See Also

Google Documentation for getAdUnitSizesByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getAdUnitSizesByStatement(dat)
## End(Not run)
```
dfp_getAllNetworks  NetworkService

Description

Provides operations for retrieving information related to the publisher’s networks. This service can be used to obtain the list of all networks that the current login has access to, or to obtain information about a specific network.

Usage

dfp_getAllNetworks(as_df = TRUE, verbose = FALSE)

Arguments

as_df  a boolean indicating whether to attempt to parse the result into a data.frame
verbose  a boolean indicating whether to print the service URL and POSTed XML

Details

getAllNetworks

Returns the list of Network objects to which the current login has access. Intended to be used without a network code in the SOAP header when the login may have more than one network associated with it.

Value

a data.frame or list containing all the elements of a getAllNetworksResponse

See Also

Google Documentation for getAllNetworks

Examples

```r
## Not run:
res <- dfp_getAllNetworks()

## End(Not run)
```
Description

Returns the Role objects that are defined for the users of the network. @return the roles defined for the user’s network

Usage

dfp_getAllRoles(as_df = TRUE, verbose = FALSE)

Arguments

as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getAllRolesResponse

See Also

Google Documentation for getAllRoles

Examples

```r
## Not run:
res <- dfp_getAllRoles()

## End(Not run)
```

---

**dfp_getAudienceSegmentsByStatement**

*getAudienceSegmentsByStatement*
dfp_getAudienceSegmentsByStatement

Description

Gets an AudienceSegmentPage of AudienceSegment objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- name
- status
- type
- size
- dataProviderName
- approvalStatus
- cost
- startDateTime
- endDateTime

Usage

dfp_getAudienceSegmentsByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getAudienceSegmentsByStatementResponse

See Also

Google Documentation for getAudienceSegmentsByStatement

Examples

## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE!'"))
res <- dfp_getAudienceSegmentsByStatement(dat)

## End(Not run)
dfp_getAvailabilityForecast

ForecastService

Description

Provides methods for estimating traffic (clicks/impressions) for line items. Forecasts can be provided for LineItem objects that exist in the system or which have not had an ID set yet.

Usage

dfp_getAvailabilityForecast(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Details

Test Network Behavior

Test networks are unable to provide forecasts that would be comparable to the production environment because forecasts require traffic history. Visit the See Also section below to proceed to Google and review the details.

getAvailabilityForecast

Gets the availability forecast for a ProspectiveLineItem. An availability forecast reports the maximum number of available units that the line item can book, and the total number of units matching the line item’s targeting.

Value

a data.frame or list containing all the elements of a getAvailabilityForecastResponse

See Also

Google Documentation for getAvailabilityForecast
Examples

```r
## Not run:
filter <- "WHERE Status='DELIVERING' LIMIT 1"
one_li <- dfp_getlineitemsByStatement(list(filterStatement=list(query=filter)))[[1]]
hypothetical_line_item <- list(lineItem=
  list(id=one_li$id,
  startDateTime=one_li$startDateTime,
  endDateTime=dfp_date_to_list(Sys.Date()+100),
  lineItemType=one_li$lineItemType,
  costType=one_li$costType,
  primaryGoal=one_li$primaryGoal,
  targeting=one_li$targeting))
request_data <- list(lineItem=hypothetical_line_item,
  forecastOptions=list(includeTargetingCriteriaBreakdown='true',
  includeContendingLineItems='true'))
dfp_getAvailabilityForecastByResult <- dfp_getAvailabilityForecast(request_data)
```

## End(Not run)

---

dfp_getAvailabilityForecastByResult

*getAvailabilityForecastByResult*

Description

Gets an AvailabilityForecast for an existing LineItem object. An availability forecast reports the maximum number of available units that the line item can be booked with, and also the total number of units matching the line item’s targeting. Only line items having type LineItemType SPONSORSHIP or [Link LineItemType STANDARD] are valid. Other types will result in [ReservationDetailsError.Reason LINE_ITEM_TYPE_NOT_ALLOWED]. Gets an AvailabilityForecast for an existing LineItem object. An availability forecast reports the maximum number of available units that the line item can be booked with, and also the total number of units matching the line item’s targeting. Only line items having type LineItemType SPONSORSHIP or LineItemType STANDARD are valid. Other types will result in ReservationDetailsError.Reason LINE_ITEM_TYPE_NOT_ALLOWED. @param lineItemId the ID of a LineItem to run the forecast on. @param forecastOptions options controlling the forecast Gets an AvailabilityForecast for an existing LineItem object. An availability forecast reports the maximum number of available units that the line item can be booked with, and also the total number of units matching the line item’s targeting. Only line items having type LineItemType SPONSORSHIP or LineItemType STANDARD are valid. Other types will result in ReservationDetailsError.Reason LINE_ITEM_TYPE_NOT_ALLOWED. @param lineItemId the ID of a LineItem to run the forecast on. @param forecastOptions options controlling the forecast

Usage

dfp_getAvailabilityForecastByResult(request_data, as_df = TRUE,
  verbose = FALSE)
Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df  a boolean indicating whether to attempt to parse the result into a data.frame
verbose  a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getAvailabilityForecastByIdResponse

See Also

Google Documentation for getAvailabilityForecastById

Examples

```r
## Not run:
filter <- "WHERE Status='DELIVERING' LIMIT 1"
one_li <- dfp_getLineItemsByStatement(list(filterStatement=list(query=filter)))[[1]]
request_data <- list(lineItemId=one_li$id,
                      forecastOptions=list(includeTargetingCriteriaBreakdown='true',
                                           includeContendingLineItems='true'))
result <- dfp_getAvailabilityForecastById(request_data)
## End(Not run)
```

Description

Gets a BaseRatePage of BaseRate objects that satisfy the given Statement query. The following fields are supported for filtering:

- rateCardId
- id
- productTemplateId

Usage

dfp_getBaseRatesByStatement(request_data, as_df = TRUE, verbose = FALSE)
dfp_getCdnConfigurationsByStatement

Arguments

- **request_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML

Value

- a data.frame or list containing all the elements of a getBaseRatesByStatementResponse

See Also

- Google Documentation for getBaseRatesByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getBaseRatesByStatement(dat)
## End(Not run)
```

Description

Gets a CdnConfigurationPage of CdnConfiguration objects that satisfy the given Statement query. Currently only CDN Configurations of type [CdnConfigurationType LIVE_STREAM_SOURCE_CONTENT] will be returned. The following fields are supported for filtering:

- **id**
- **name**

Usage

```r
dfp_getCdnConfigurationsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- **request_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML
dfp_getCmsMetadataKeysByStatement

Value

A data.frame or list containing all the elements of a getCdnConfigurationsByStatementResponse

See Also

Google Documentation for getCdnConfigurationsByStatement

Examples

```r
# Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getCdnConfigurationsByStatement(dat)

# End(Not run)
```

dfp_getCmsMetadataKeysByStatement

CmsMetadataService

Description

Provides methods for querying CMS metadata keys and values. A CMS metadata value corresponds to one key value pair ingested from a publisher’s CMS and is used to target all the content with which it is associated in the CMS.

Usage

dfp_getCmsMetadataKeysByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- `request_data`: A list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: A boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: A boolean indicating whether to print the service URL and POSTed XML

Details

getCmsMetadataKeysByStatement

Returns a page of CmsMetadataKeys matching the specified Statement. The following fields are supported for filtering:

- `id`
- `cmsKey`
dfp_getCmsMetadataValuesByStatement

Value

a data.frame or list containing all the elements of a getCmsMetadataKeysByStatementResponse

See Also

Google Documentation for getCmsMetadataKeysByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getCmsMetadataKeysByStatement(dat)
## End(Not run)
```

---

dfp_getCmsMetadataValuesByStatement

getCmsMetadataValuesByStatement

Description

Returns a page of CmsMetadataValues matching the specified Statement. The following fields are supported for filtering:

- id
- cmsValue
- cmsKey
- keyValueMemberContent

Usage

dfp_getCmsMetadataValuesByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getCmsMetadataValuesByStatementResponse
See Also

Google Documentation for getCmsMetadataValuesByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getCmsMetadataValuesByStatement(dat)

## End(Not run)
```

dfp_getCompaniesByStatement

getCompaniesByStatement

Description

Gets a CompanyPage of Company objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- name
- type
- lastModifiedDateTime

Usage

dfp_getCompaniesByStatement(request_data, as_df = TRUE,
verbose = FALSE)

Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Value

A data.frame or list containing all the elements of a getCompaniesByStatementResponse

See Also

Google Documentation for getCompaniesByStatement
dfp_getContactsByStatement

getContactsByStatement

Description

Gets a ContactPage of Contact objects that satisfy the given Statement query. The following fields are supported for filtering:

- name
- email
- id
- comment
- companyId
- title
- cellPhone
- workPhone
- faxPhone
- status

Usage

dfp_getContactsByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df a boolean indicating whether to attempt to parse the result into a data.frame

verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getContactsByStatementResponse

See Also

Google Documentation for getContactsByStatement
dfp_getContentBundlesByStatement

Examples

## Not run:
```r
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getContactsByStatement(dat)
```

## End(Not run)

### dfp_getContentBundlesByStatement

#### getContentBundlesByStatement

**Description**

Gets a ContentBundlePage of ContentBundle objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- name
- status

**Usage**

```
dfp_getContentBundlesByStatement(request_data, as_df = TRUE,
    verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a getContentBundlesByStatementResponse

**See Also**

Google Documentation for getContentBundlesByStatement

**Examples**

## Not run:
```r
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getContentBundlesByStatement(dat)
```

## End(Not run)
dfp_getContentByStatement

ContentService

Description

Service for retrieving Content. Content entities can be targeted in video LineItems. You can query for content that belongs to a particular category or has assigned metadata. Categories and metadata for Content are stored in DFP as CustomCriteria. For example, to find all Content that is "genre=comedy", you would:

• Retrieve the custom targeting key corresponding to "genre" using CustomTargetingService#getCustomTargetingKeysByStatement
• Using the CustomTargetingService#getCustomTargetingValuesByStatement method and a filter like "WHERE customTargetingKeyId = :genreKeyId and name = 'comedy'", retrieve the ID for the "comedy" custom targeting value.
• Call #getContentByStatementAndCustomTargetingValue with a filter like "WHERE status = 'ACTIVE'" and the ID of the custom targeting value from step 2.

Usage

dfp_getContentByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Details

g getContentByStatement

Gets a ContentPage of Content objects that satisfy the given [ @link Statement query ]. The following fields are supported for filtering:

• id
• status
• name
• lastModifiedDateTime
• lastDaiIngestDateTime
• daiIngestStatus

Value

a data.frame or list containing all the elements of a getContentByStatementResponse
dfp_getContentByStatementAndCustomTargetingValue

See Also
Google Documentation for getContentByStatement

Examples
```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getContentByStatement(dat)
```

Description
Gets a ContentPage of Content objects that satisfy the given Statement query. Additionally, filters on the given value ID and key ID that the value belongs to. The following fields are supported for filtering:

- id
- status
- name
- lastModifedDateTime

Usage
```r
dfp_getContentByStatementAndCustomTargetingValue(as_df = TRUE, verbose = FALSE)
```

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>as_df</td>
<td>a boolean indicating whether to attempt to parse the result into a data.frame</td>
</tr>
<tr>
<td>verbose</td>
<td>a boolean indicating whether to print the service URL and POSTed XML</td>
</tr>
</tbody>
</table>

Value
a data.frame or list containing all the elements of a getContentByStatementAndCustomTargetingValueResponse

See Also
Google Documentation for getContentByStatementAndCustomTargetingValue
Examples

```r
## Not run:
res <- dfp_getContentByStatementAndCustomTargetingValue()

## End(Not run)
```

dfp_getCreativesByStatement

```
getCreativesByStatement
```

Description

Gets a CreativePage of Creative objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- name
- advertiserId
- width
- height
- lastModifiedDateTime

Usage

```r
dfp_getCreativesByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Value

A `data.frame` or `list` containing all the elements of a `getCreativesByStatementResponse`

See Also

- [Google Documentation for `getCreativesByStatement`](#)
dfp_getCreativeSetsByStatement

Examples

## Not run:
```r
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getCreativesByStatement(dat)
```
## End(Not run)

---

**Description**

Gets a CreativeSetPage of CreativeSet objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- name
- masterCreativeId
- lastModifiedDateTime

**Usage**

```r
dfp_getCreativeSetsByStatement(request_data, as_df = TRUE,
   verbose = FALSE)
```

**Arguments**

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

a `data.frame` or list containing all the elements of a `getCreativeSetsByStatementResponse`

**See Also**

- Google Documentation for `getCreativeSetsByStatement`

**Examples**

## Not run:
```r
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getCreativeSetsByStatement(dat)
```
## End(Not run)
dfp_getCreativeTemplatesByStatement

CreatableTemplateService

Description
Provides methods for retrieving CreativeTemplate objects.

Usage

dfp_getCreativeTemplatesByStatement(request_data, as_df = FALSE,
   verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request
   (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Details

genericCreativeTemplatesByStatement

Gets a CreativeTemplatePage of CreativeTemplate objects that satisfy the given Statement query.

The following fields are supported for filtering:

• id
• name
• type
• status

Value

a data.frame or list containing all the elements of a getCreativeTemplatesByStatementResponse

See Also

Google Documentation for getCreativeTemplatesByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getCreativeTemplatesByStatement(dat)

## End(Not run)
```
dfp_getCreativeWrappersByStatement

Description

Gets a CreativeWrapperPage of CreativeWrapper objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- labelId
- status
- ordering

Usage

```r
dfp_getCreativeWrappersByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- `request_data` a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getCreativeWrappersByStatementResponse

See Also

Google Documentation for getCreativeWrappersByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getCreativeWrappersByStatement(dat)

## End(Not run)
```
**dfp_getCurrentNetwork**  \hspace{1cm} **getCurrentNetwork**

### Description

Returns the current network for which requests are being made. Returns the current network for which requests are being made. @return the network for which the user is currently making the request

### Usage

```r
dfp_getCurrentNetwork(as_df = TRUE, verbose = FALSE)
```

### Arguments

- `as_df`  a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`  a boolean indicating whether to print the service URL and POSTed XML

### Value

a `data.frame` or list containing all the elements of a `getCurrentNetworkResponse`

### See Also

[Google Documentation for getCurrentNetwork](https://example.com)

### Examples

```r
## Not run:
res <- dfp_getCurrentNetwork()

## End(Not run)
```

---

**dfp_getCurrentUser**  \hspace{1cm} **getCurrentUser**

### Description

Returns the current User. Returns the current User. @return the current user

### Usage

```r
dfp_getCurrentUser(as_df = TRUE, verbose = FALSE)
```

---
dfp_getCustomFieldOption

Arguments

- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

- a data.frame or list containing all the elements of a getCurrentUserResponse

See Also

- Google Documentation for getCurrentUser

Examples

```r
## Not run:
res <- dfp_getCurrentUser()
## End(Not run)
```

dfp_getCustomFieldOption

getCustomFieldOption

Description

Returns the CustomFieldOption uniquely identified by the given ID.

Usage

```r
dfp_getCustomFieldOption(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

- a data.frame or list containing all the elements of a getCustomFieldOptionResponse

See Also

- Google Documentation for getCustomFieldOption
### dfp_getCustomFieldsByStatement

**getCustomFieldsByStatement**

**Description**

Gets a CustomFieldPage of CustomField objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- entityType
- name
- isActive
- visibility

**Usage**

```r
dfp_getCustomFieldsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a getCustomFieldsByStatementResponse

**See Also**

[Google Documentation for getCustomFieldsByStatement](#)

**Examples**

```r
## Not run:
res <- dfp_getCustomFieldOption(request_data)
## End(Not run)
```

```r
dfp_getCustomFieldsByStatement
```

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getCustomFieldsByStatement(dat)
## End(Not run)
```
Description

Gets a CustomTargetingKeyPage of CustomTargetingKey objects that satisfy the given Statement query. The following fields are supported for filtering:

- NA
- id
- name
- displayName
- type

Usage

dfp_getCustomTargetingKeysByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getCustomTargetingKeysByStatementResponse

See Also

Google Documentation for getCustomTargetingKeysByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getCustomTargetingKeysByStatement(dat)
## End(Not run)
```
**Description**

Gets a CustomTargetingValuePage of CustomTargetingValue objects that satisfy the given Statement query. The WHERE clause in the Statement query must always contain CustomTargetingValue customTargetingKeyId as one of its columns in a way that it is AND'ed with the rest of the query. So, if you want to retrieve values for a known set of key ids, valid Statement query would look like: "WHERE customTargetingKeyId IN ('17','18','19')" retrieves all values that are associated with keys having ids 17, 18, 19. "WHERE customTargetingKeyId = '17' AND name = 'red'" retrieves values that are associated with keys having id 17 and value name is 'red'. The following fields are supported for filtering:

- id
- customTargetingKeyId
- name
- displayName
- matchType

**Usage**

```r
dfp_getCustomTargetingValuesByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a getCustomTargetingValuesByStatementResponse

**See Also**

Google Documentation for getCustomTargetingValuesByStatement
dfp_getDaiAuthenticationKeysByStatement

## Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getCustomTargetingValuesByStatement(dat)

## End(Not run)
```

### Description

Gets a DaiAuthenticationKeyPage of DaiAuthenticationKey objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- status
- name

### Usage

```r
dfp_getDaiAuthenticationKeysByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

### Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

### Value

A data.frame or list containing all the elements of a getDaiAuthenticationKeysByStatementResponse

### See Also

[Google Documentation for getDaiAuthenticationKeysByStatement](#)

### Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getDaiAuthenticationKeysByStatement(dat)

## End(Not run)
```
dfp_getDeliveryForecast

getDeliveryForecast

Description

Gets the delivery forecast for a list of ProspectiveLineItem objects in a single delivery simulation with line items potentially contending with each other. A delivery forecast reports the number of units that will be delivered to each line item given the line item goals and contentions from other line items. Gets the delivery forecast for a list of ProspectiveLineItem objects in a single delivery simulation with line items potentially contending with each other. A delivery forecast reports the number of units that will be delivered to each line item given the line item goals and contentions from other line items. @param lineItems line items to be forecasted for delivery @param forecastOptions options controlling the forecast

Usage

dfp_getDeliveryForecast(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getDeliveryForecastResponse

See Also

Google Documentation for getDeliveryForecast

Examples

## Not run:
filter <- "WHERE Status='DELIVERING' LIMIT 1"
one_li <- dfp_getLineItemsByStatement(list(filterStatement=list(query=filter)))[[1]]
hypothetical_line_item <- list(lineItem=
  list(id=one_li$id,
   startDateTime=one_li$startTime,
   endDateTime=Sys.Date()+100),
Description

Gets the delivery forecast for a list of existing LineItem objects in a single delivery simulation. A delivery forecast reports the number of units that will be delivered to each line item given the line item goals and contentions from other line items. Gets the delivery forecast for a list of existing LineItem objects in a single delivery simulation. A delivery forecast reports the number of units that will be delivered to each line item given the line item goals and contentions from other line items. @param lineItemIds the IDs of line items to be forecasted for delivery @param forecastOptions options controlling the forecast

Usage

dfp_getDeliveryForecastByIds(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getDeliveryForecastByIdsResponse

See Also

Google Documentation for getDeliveryForecastByIds
dfp_getExchangeRatesByStatement

getExchangeRatesByStatement

Description

Gets a ExchangeRatePage of ExchangeRate objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- currencyCode
- refreshRate
- direction
- exchangeRate

Usage

dfp_getExchangeRatesByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

A data.frame or list containing all the elements of a getExchangeRatesByStatementResponse

See Also

Google Documentation for getExchangeRatesByStatement
dfp_getLabelsByStatement

Description

Gets a LabelPage of Label objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- type
- name
- description
- isActive

Usage

dfp_getLabelsByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getLabelsByStatementResponse

See Also

Google Documentation for getLabelsByStatement

Examples

## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getExchangeRatesByStatement(dat)

## End(Not run)
dfp_getLineItemCreativeAssociationsByStatement

**Description**

Gets a LineItemCreativeAssociationPage of LineItemCreativeAssociation objects that satisfy the given Statement query. The following fields are supported for filtering:

- creativeId
- manualCreativeRotationWeight
- destinationUrl
- lineItemId
- status
- lastModifiedDateTime

**Usage**

```r
dfp_getLineItemCreativeAssociationsByStatement(request_data, 
as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data` a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a `getLineItemCreativeAssociationsByStatementResponse`

**See Also**

[Google Documentation for `getLineItemCreativeAssociationsByStatement`]

**Examples**

```r
## Not run:
 dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
 res <- dfp_getLineItemCreativeAssociationsByStatement(dat)

## End(Not run)
```
Description

Gets a LineItemPage of LineItem objects that satisfy the given Statement query. The following fields are supported for filtering:

- CostType
- CreationDateTime
- DeliveryRateType
- EndDateTime
- ExternalId
- Id
- IsMissingCreatives
- IsSetTopBoxEnabled
- LastModifiedDateTime
- LineItemType
- Name
- OrderId
- StartDateTime
- Status
- Targeting
- UnitsBought

Usage

dfp_getLineItemsByStatement(request_data, as_df = FALSE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df a boolean indicating whether to attempt to parse the result into a data.frame

verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getLineItemsByStatementResponse
dfp_getLineItemTemplatesByStatement

LineItemTemplateService

Description

Provides methods for creating, updating and retrieving LineItemTemplate objects.

Usage

```r
dfp_getLineItemTemplatesByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Details

`dfp_getLineItemTemplatesByStatement`

Gets a LineItemTemplatePage of LineItemTemplate objects that satisfy the given Statement query. The following fields are supported for filtering:

- id

Value

a data.frame or list containing all the elements of a getLineItemTemplatesByStatementResponse

See Also

Google Documentation for dfp_getLineItemTemplatesByStatement
**dfp_getLiveStreamEventsByStatement**  

**Examples**

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getLineItemTemplatesByStatement(dat)

## End(Not run)
```

---

**dfp_getLiveStreamEventsByStatement**

getLiveStreamEventsByStatement

**Description**

Gets a LiveStreamEventPage of LiveStreamEvent objects that satisfy the given Statement query. The following fields are supported for filtering:

- id

**Usage**

```r
dfp_getLiveStreamEventsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a getLiveStreamEventsByStatementResponse

**See Also**

Google Documentation for getLiveStreamEventsByStatement

**Examples**

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getLiveStreamEventsByStatement(dat)

## End(Not run)
```
dfp_getMarketplaceCommentsByStatement

*getMarketplaceCommentsByStatement*

**Description**

Gets a MarketplaceCommentPage of MarketplaceComment objects that satisfy the given Statement query. This method only returns comments already sent to Marketplace, local draft ProposalMarketplaceInfo marketplaceComment are not included. The following fields are supported for filtering:

- proposalId

**Usage**

```r
dfp_getMarketplaceCommentsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data` a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a getMarketplaceCommentsByStatementResponse

**See Also**

[Google Documentation for getMarketplaceCommentsByStatement](https://www.google.com)

**Examples**

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getMarketplaceCommentsByStatement(dat)

## End(Not run)
```
**Description**

Gets a `MobileApplicationPage mobileApplicationPage` of `MobileApplication mobile applications` that satisfy the given `Statement`. The following fields are supported for filtering:

- `id`
- `displayName`
- `appStore`
- `appStoreId`
- `NA`
- `isArchived`

**Usage**

```r
dfp_getMobileApplicationsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

A `data.frame` or list containing all the elements of a `getMobileApplicationsByStatementResponse`

**See Also**

- [Google Documentation for `getMobileApplicationsByStatement`](#)

**Examples**

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getMobileApplicationsByStatement(dat)

## End(Not run)
```
dfp_getNativeStylesByStatement

Description

Gets a NativeStylePage NativeStylePage of NativeStyle objects that satisfy the given Statement. The following fields are supported for filtering:

- id
- name

Usage

dfp_getNativeStylesByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getNativeStylesByStatementResponse

See Also

Google Documentation for getNativeStylesByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getNativeStylesByStatement(dat)

## End(Not run)
```
**Description**

Gets an OrderPage of Order objects that satisfy the given Statement query. The following fields are supported for filtering:

- advertiserId
- endDateTime
- id
- name
- salespersonId
- startDateTime
- status
- traffickerId
- lastModifiedDateTime

**Usage**

```r
dfp_getOrdersByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

- a data.frame or list containing all the elements of a getOrdersByStatementResponse

**See Also**

Google Documentation for getOrdersByStatement

**Examples**

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getOrdersByStatement(dat)

## End(Not run)
```
Description

Gets a PackagePage of Package objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- name
- proposalId
- productPackageId
- isArchived
- lastModifiedDateTime

Usage

dfp_getPackagesByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getPackagesByStatementResponse

See Also

Google Documentation for getPackagesByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getPackagesByStatement(dat)
## End(Not run)
```
Description

Gets a PlacementPage of Placement objects that satisfy the given Statement query. The following fields are supported for filtering:

- description
- id
- name
- placementCode
- status
- lastModifiedDateTime

Usage

```r
dfp_getPlacementsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- `request_data` a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

Value

a `data.frame` or list containing all the elements of a `getPlacementsByStatementResponse`

See Also

Google Documentation for `getPlacementsByStatement`

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getPlacementsByStatement(dat)
## End(Not run)
```
Description

Gets a PremiumRatePage of PremiumRate objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- rateCardId
- pricingMethod

Usage

dfp_getPremiumRatesByStatement(request_data, as_df = TRUE,
    verbose = FALSE)

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

- a data.frame or list containing all the elements of a getPremiumRatesByStatementResponse

See Also

Google Documentation for getPremiumRatesByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getPremiumRatesByStatement(dat)

## End(Not run)
```
**dfp_getPreviewUrl**

---

### dfp_getPreviewUrl  getPreviewUrl

**Description**

Returns an insite preview URL that references the specified site URL with the specified creative from the association served to it. For Creative Set previewing you may specify the master creative Id.

**Usage**

```
dfp_getPreviewUrl(as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `as_df` a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Value**

A `data.frame` or list containing all the elements of a `getPreviewUrlResponse`

**See Also**

- [Google Documentation for getPreviewUrl](#)

**Examples**

```r
## Not run:
res <- dfp_getPreviewUrl()

## End(Not run)
```

---

**dfp_getPreviewUrlsForNativeStyles**

---

### dfp_getPreviewUrlsForNativeStyles  getPreviewUrlsForNativeStyles

**Description**

Returns a list of URLs that reference the specified site URL with the specified creative from the association served to it. For Creative Set previewing you may specify the master creative Id. Each URL corresponds to one available native style for previewing the specified creative.

**Usage**

```
dfp_getPreviewUrlsForNativeStyles(as_df = TRUE, verbose = FALSE)
```

---
**Arguments**

- `as_df` a boolean indicating whether to attempt to parse the result into a data frame
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Value**

A data frame or list containing all the elements of a getPreviewUrlsForNativeStylesResponse

**See Also**

Google Documentation for getPreviewUrlsForNativeStyles

**Examples**

```r
## Not run:
res <- dfp_getPreviewUrlsForNativeStyles()
## End(Not run)
```

---

**Description**

Gets a ProductPackageItemPage of ProductPackageItem objects that satisfy the filtering criteria specified by given Statement query. The following fields are supported for filtering:

- id
- productPackageId
- productId
- productTemplateId
- mandatory
- archived

**Usage**

```r
dfp_getProductPackageItemsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data` a list or data frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a data frame
- `verbose` a boolean indicating whether to print the service URL and POSTed XML
dfp_getProductPackagesByStatement

Value

a data.frame or list containing all the elements of a getProductPackageItemsByStatementResponse

See Also

Google Documentation for getProductPackageItemsByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getProductPackageItemsByStatement(dat)

## End(Not run)
```

dfp_getProductPackagesByStatement

### Description

Gets a ProductPackagePage of ProductPackage objects that satisfy the filtering criteria specified by given Statement query. The following fields are supported for filtering:

- id
- name
- notes
- status
- isArchived
- lastModifiedDateTime

### Usage

dfp_getProductPackagesByStatement(request_data, as_df = TRUE, verbose = FALSE)

### Arguments

- **request_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML
dfp_getProductsByStatement

Value

A data.frame or list containing all the elements of a getProductPackagesByStatementResponse.

See Also

Google Documentation for getProductPackagesByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getProductPackagesByStatement(dat)

## End(Not run)
```

dfp_getProductsByStatement

 ProductService

Description

Provides methods for updating and retrieving Product objects. A Product represents a line item proposal. Products are generated from ProductTemplate product templates on a periodic basis using the product template’s attributes. Products are typically used by inventory managers to restrict what salespeople can sell. To use this service, you need to have the new sales management solution enabled on your network. If you do not see a "Sales" tab in a href="https://www.google.com/dfp">DoubleClick for Publishers (DFP)</a>, you will not be able to use this service.

Usage

```
dfp_getProductsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Details

getProductsByStatement

Gets a ProductPage of Product objects that satisfy the criteria specified by given Statement query. When using sales management, the following fields are supported for filtering and/or sorting.

- `rateCardId`
- `status`
```
# Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getProductsByStatement(dat)
```

**Value**

A data.frame or list containing all the elements of a getProductsByStatementResponse

**See Also**

Google Documentation for getProductsByStatement

**Examples**

```
# Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getProductsByStatement(dat)
```

**Description**

Gets a ProductTemplatePage of ProductTemplate objects that satisfy the filtering criteria specified by given Statement query. The following fields are supported for filtering:

- id
- name
- nameMacro
- description
- status
- lastModifiedDateTime
- lineItemType
- productType
- rateType
Usage

dfp_getProductTemplatesByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df  a boolean indicating whether to attempt to parse the result into a data.frame
verbose  a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getProductTemplatesByStatementResponse

See Also

Google Documentation for getProductTemplatesByStatement

Examples

## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getProductTemplatesByStatement(dat)

## End(Not run)

dfp_getProposalLineItemsByStatement

getProposalLineItemsByStatement

Description

 Gets a ProposalLineItemPage of ProposalLineItem objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- name
- proposalId
- startDateDateTime
- endDateDateTime
- isArchived
- lastModifiedDateTime
- useThirdPartyAdServerFromProposal Only applicable for non-programmatic proposal line items using sales management


• thirdPartyAdServerId Only applicable for non-programmatic proposal line items using sales management
• customThirdPartyAdServerName Only applicable for non-programmatic proposal line items using sales management
• isProgrammatic

Usage

dfp_getProposalLineItemsByStatement(request_data, as_df = TRUE,
verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request
(XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getProposalLineItemsByStatementResponse

See Also

Google Documentation for getProposalLineItemsByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getProposalLineItemsByStatement(dat)
## End(Not run)
```

dfp_getProposalsByStatement

Description

Gets a ProposalPage of Proposal objects that satisfy the given Statement query. The following fields are supported for filtering:

• id
• dfpOrderId
• name
dfp_getProposalsByStatement

- status
- isArchived
- approvalStatus Only applicable for proposals using sales management
- lastModifiedDateTime
- thirdPartyAdServerId Only applicable for non-programmatic proposals using sales management
- customThirdPartyAdServerName Only applicable for non-programmatic proposals using sales management
- hasOfflineErrors
- isProgrammatic
- negotiationStatus Only applicable for programmatic proposals

Usage

```
dfp_getProposalsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getProposalsByStatementResponse

See Also

Google Documentation for getProposalsByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getProposalsByStatement(dat)

## End(Not run)
```
dfp_getRateCardsByStatement

getRateCardsByStatement

Description

Gets a RateCardPage of RateCard objects that satisfy the given Statement query. The following fields are supported for filtering:

- ForMarketplace
- Id
- LastModifiedDateTime
- Name
- Status

Usage

dfp_getRateCardsByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df: a boolean indicating whether to attempt to parse the result into a data.frame

verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a getRateCardsByStatementResponse

See Also

Google Documentation for getRateCardsByStatement

Examples

## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getRateCardsByStatement(dat)

## End(Not run)
Description

Provides methods for retrieving and updating ReconciliationLineItemReport objects.

Usage

dfp_getReconciliationLineItemReportsByStatement(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Details

getReconciliationLineItemReportsByStatement

Gets a ReconciliationLineItemReportPage of ReconciliationLineItemReport objects that satisfy the given Statement query. The following fields are supported for filtering:

• id
• reconciliationReportId
• orderId
• proposalId
• lineItemId
• proposalLineItemId

Value

a data.frame or list containing all the elements of a getReconciliationLineItemReportsByStatementResponse

See Also

Google Documentation for getReconciliationLineItemReportsByStatement
dfp_getReconciliationOrderReportsByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getReconciliationLineItemReportsByStatement(dat)
```

## End(Not run)

---

**dfp_getReconciliationOrderReportsByStatement**

*ReconciliationOrderReportService*

**Description**

Provides methods for retrieving, reconciling, and reverting ReconciliationOrderReport objects.

**Usage**

```
dfp_getReconciliationOrderReportsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Details**

`getReconciliationOrderReportsByStatement`

Gets an ReconciliationOrderReportPage of ReconciliationOrderReport objects that satisfy the given Statement query. The following fields are supported for filtering:

- reconciliationReportId
- id
- orderId
- proposalId
- status
- submissionDateTime
- submitterId

**Value**

a data.frame or list containing all the elements of a `getReconciliationOrderReportsByStatementResponse`
See Also

Google Documentation for `getReconciliationOrderReportsByStatement`

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getReconciliationOrderReportsByStatement(dat)

## End(Not run)
```

---

`dfp_getReconciliationReportRowsByStatement`

*ReconciliationReportRowService*

Description

Provides methods for retrieving and updating the `ReconciliationReportRow` objects.

Usage

```r
dfp_getReconciliationReportRowsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- **request_data**: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a `data.frame`
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML

Details

getReconciliationReportRowsByStatement

Gets a `ReconciliationReportRowPage` of `ReconciliationReportRow` objects that satisfy the given Statement query. The following fields are supported for filtering:

- reconciliationReportId
- advertiserId
- orderId
- lineItemId
- proposalLineItemId
- creativeId
- lineItemCostType
- dfpClicks
dfp_getReconciliationReportsByStatement

- dfpImpressions
- dfpLineItemDays
- thirdPartyClicks
- thirdPartyImpressions
- thirdPartyLineItemDays
- manualClicks
- manualImpressions
- manualLineItemDays
- reconciledClicks
- reconciledImpressions
- reconciledLineItemDays

Value

A data.frame or list containing all the elements of a getReconciliationReportRowsByStatementResponse

See Also

Google Documentation for getReconciliationReportRowsByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list(query="WHERE status='ACTIVE'"))
res <- dfp_getReconciliationReportRowsByStatement(dat)

## End(Not run)
```

dfp_getReconciliationReportsByStatement

*ReconciliationReportService*

Description

Provides methods for retrieving, submitting and reverting the ReconciliationReport objects. A ReconciliationReport is a group of ReconciliationReportRow objects.

Usage

dfp_getReconciliationReportsByStatement(request_data, as_df = TRUE, verbose = FALSE)
Arguments
request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df  a boolean indicating whether to attempt to parse the result into a data.frame
verbose  a boolean indicating whether to print the service URL and POSTed XML

Details
getReconciliationReportsByStatement
Gets an ReconciliationReportPage of ReconciliationReport objects that satisfy the given Statement query. The following fields are supported for filtering.
• id
• status
• startDate

Value
a data.frame or list containing all the elements of a getReconciliationReportsByStatementResponse

See Also
Google Documentation for getReconciliationReportsByStatement

Examples
```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getReconciliationReportsByStatement(dat)
```

Description
Provides methods for executing a ReportJob and retrieving performance and statistics about ad campaigns, networks, inventory and sales. Follow the steps outlined below:
• Create the ReportJob object by invoking ReportService#runReportJob.
• Continue to poll the ReportJob object until the ReportJob#reportJobStatus field is equal to ReportJobStatus#COMPLETED or ReportJobStatus#FAILED.
dfp_getReportDownloadURL

- If successful, fetch the URL for downloading the report by invoking ReportService#getReportDownloadURL.

<h4>Test network behavior</h4> The networks created using NetworkService#makeTestNetwork are unable to provide reports that would be comparable to the production environment because reports require traffic history. In the test networks, reports will consistently return no data for all reports.

Usage

dfp_getReportDownloadURL(request_data, as_df = FALSE, verbose = FALSE)

Arguments

- **request_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML

Details

gReportDownloadURL

Returns the URL at which the report file can be downloaded. The report will be generated as a gzip archive, containing the report file itself.

Value

a data.frame or list containing all the elements of a getReportDownloadURLResponse

See Also

Google Documentation for getReportDownloadURL

Examples

```r
## Not run:
request_data <- list(reportJob=
  list(reportQuery=
    list(dimensions='MONTH_AND_YEAR',
        dimensions='AD_UNIT_ID',
        adUnitView='FLAT',
        columns='AD_SERVER_CLICKS',
        dateRangeType='LAST_WEEK')))

# the result is a list and most importantly the ID is included for checking its status
dfp_runReportJob_result <- dfp_runReportJob(request_data)

# only run after the status is "COMPLETED"
request_data <- list(reportJobId=dfp_runReportJob_result$id, exportFormat='CSV_DUMP')
dfp_getReportDownloadURL_result <- dfp_getReportDownloadURL(request_data)

## End(Not run)
```
**dfp_getReportDownloadUrlWithOptions**

Description

Returns the URL at which the report file can be downloaded, and allows for customization of the downloaded report. By default, the report will be generated as a gzip archive, containing the report file itself. This can be changed by setting ReportDownloadOptions useGzipCompression to false.

Usage

```r
dfp_getReportDownloadUrlWithOptions(request_data, as_df = FALSE, verbose = FALSE)
```

Arguments

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Value

A `data.frame` or list containing all the elements of a `getReportDownloadUrlWithOptionsResponse`

See Also

Google Documentation for `getReportDownloadUrlWithOptions`

Examples

```r
## Not run:
res <- dfp_getReportDownloadUrlWithOptions(request_data)

## End(Not run)```
dfp_getReportJobStatus

getReportJobStatus

Description

Returns the ReportJobStatus of the report job with the specified ID. Returns the ReportJobStatus of the report job with the specified ID. Returns the ReportJobStatus of the report job with the specified ID.

Usage

dfp_getReportJobStatus(request_data, as_df = FALSE, verbose = FALSE)

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

A data.frame or list containing all the elements of a getReportJobStatusResponse

See Also

Google Documentation for getReportJobStatus

Examples

```r
## Not run:
request_data <- list(reportJob=
  list(reportQuery=
    list(dimensions='MONTH_AND_YEAR',
      dimensions='AD_UNIT_ID',
      adUnitView='FLAT',
      columns='AD_SERVER_CLICKS',
      dateRangeType='LAST_WEEK')))

# the result is a list and most importantly the ID is included for checking its status
dfp_runReportJob_result <- dfp_runReportJob(request_data)

request_data <- list(reportJobId=dfp_runReportJob_result$id)
dfp_getReportJobStatus_result <- dfp_getReportJobStatus(request_data)
dfp_getReportJobStatus_result

# a simple while loop can keep checking a long running request until ready
```
dfp_getSavedQueriesByStatement

getSavedQueriesByStatement

**Description**

Retrieves a page of the saved queries either created by or shared with the current user. Each Saved-Query in the page, if it is compatible with the current API version, will contain a ReportQuery object which can be optionally modified and used to create a ReportJob. This can then be passed to ReportService runReportJob. The following fields are supported for filtering:

- id
- name

**Usage**

dfp_getSavedQueriesByStatement(request_data, as_df = FALSE, verbose = FALSE)

**Arguments**

- `request_data` a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Value**

- a data.frame or list containing all the elements of a getSavedQueriesByStatementResponse

**See Also**

Google Documentation for getSavedQueriesByStatement
**dfp_getSuggestedAdUnitsByStatement**

### Description

This service provides operations for retrieving and approving SuggestedAdUnit objects. Publishers may create ad tags that lack a corresponding ad unit defined in DFP, in order to gather information about potential ads without needing to create dummy ad units and make them available for targeting in line items. Any undefined ad unit to receive more than ten serving requests in the past week is treated as a 'suggested ad unit'. These can be queried by the client and selectively approved. Approval causes a new ad unit to be created based on the suggested ad unit. Unapproved suggested ad units cease to exist whenever their corresponding ad tag has been served fewer than ten times in the past seven days. This service is only available to Premium publishers. Before use, suggested ad units must be enabled for the client’s network. This can be done in the UI: in the Inventory tab, click “Network settings” in the left-hand panel, then enable the checkbox “Get suggestions for new ad units.” If suggested ad units are not enabled, then #getSuggestedAdUnitsByStatement will always return an empty page.

### Usage

```r
dfp_getSuggestedAdUnitsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

### Arguments

- **request_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML
**Details**

getSuggestedAdUnitsByStatement

Gets a SuggestedAdUnitPage of SuggestedAdUnit objects that satisfy the filter query. There is a system-enforced limit of 1000 on the number of suggested ad units that are suggested at any one time. **Note:** After API version 201311, the id field will only be numerical.

- id
- numRequests

**Value**

a data.frame or list containing all the elements of a getSuggestedAdUnitsByStatementResponse

**See Also**

Google Documentation for getSuggestedAdUnitsByStatement

**Examples**

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getSuggestedAdUnitsByStatement(dat)
## End(Not run)
```

**dfp_getTargetingPresetsByStatement**

*TargetingPresetService*

**Description**

Service for interacting with Targeting Presets.

**Usage**

```r
dfp_getTargetingPresetsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data` a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose` a boolean indicating whether to print the service URL and POSTed XML
Details

getTargetingPresetsByStatement

Gets a TargetingPresetPage of TargetingPreset objects that satisfy the given Statement query. The following fields are supported for filtering:

• id
• name

Value

A `data.frame` or list containing all the elements of a `getTargetingPresetsByStatementResponse`

See Also

Google Documentation for `getTargetingPresetsByStatement`

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getTargetingPresetsByStatement(dat)
## End(Not run)
```

Description

`getTeamsByStatement`

Description

Gets a TeamPage of Team objects that satisfy the given Statement query. The following fields are supported for filtering:

• id
• name
• description

Usage

`dfp_getTeamsByStatement(request_data, as_df = TRUE, verbose = FALSE)`

Arguments

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML
**Value**

A data frame or list containing all the elements of a getTeamsByStatementResponse.

**See Also**

Google Documentation for getTeamsByStatement.

**Examples**

```r
## Not run:
dat <- list(filterStatement=list('query'='WHERE status='ACTIVE'))
res <- dfp_getTeamsByStatement(dat)

## End(Not run)
```

---

**dfp_getTrafficAdjustmentsByStatement**

`AdjustmentService`

**Description**

Provides methods for creating, updating and retrieving Adjustment objects.

**Usage**

```r
dfp_getTrafficAdjustmentsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: A list or data frame of data elements to be formatted for a SOAP request (XML format, but passed as character string).
- `as_df`: A boolean indicating whether to attempt to parse the result into a data frame.
- `verbose`: A boolean indicating whether to print the service URL and POSTed XML.

**Details**

Adjustments are used to adjust a particular ad unit for forecasting. For example, you might have a manual adjustment for an inventory unit that will be seeing a spike for a movie premiere coming up. Or you may have a historical adjustment to tell forecasting that you have a seasonal trend coming up and you want Christmas this year to look like Christmas last year plus five percent.

**getTrafficAdjustmentsByStatement**

Returns a TrafficForecastAdjustmentPage of all TrafficForecastAdjustments that satisfy the given Statement query. The following fields are supported for filtering:

- id
- lastModifiedDateTime
**dfp_getUsersByStatement**

**Value**

A data frame or list containing all the elements of a `getTrafficAdjustmentsByStatementResponse`

**See Also**

[Google Documentation for `getTrafficAdjustmentsByStatement`]

**Examples**

```r
## Not run:
dat <- list(filterStatement=list(query="WHERE status='ACTIVE'"))
res <- dfp_getTrafficAdjustmentsByStatement(dat)

## End(Not run)
```

---

**dfp_getUsersByStatement**

*getUsersByStatement*

**Description**

Gets a UserPage of User objects that satisfy the given Statement query. The following fields are supported for filtering:

- email
- id
- name
- roleId
- rolename
- status

**Usage**

`dfp_getUsersByStatement(request_data, as_df = TRUE, verbose = FALSE)`

**Arguments**

- `request_data` - A list or data frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` - A boolean indicating whether to attempt to parse the result into a data frame
- `verbose` - A boolean indicating whether to print the service URL and POSTed XML

**Value**

A data frame or list containing all the elements of a `getUsersByStatementResponse`
**_dfp_getUserTeamAssociationsByStatement**

**See Also**

Google Documentation for getUsersByStatement

**Examples**

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getUsersByStatement(dat)
## End(Not run)
```

**dfp_getUserTeamAssociationsByStatement**

**getUserTeamAssociationsByStatement**

**Description**

Gets a UserTeamAssociationPage of UserTeamAssociation objects that satisfy the given Statement query. The following fields are supported for filtering:

- `userId`
- `teamId`

**Usage**

```r
dfp_getUserTeamAssociationsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

a `data.frame` or `list` containing all the elements of a `getUserTeamAssociationsByStatementResponse`

**See Also**

Google Documentation for `getUserTeamAssociationsByStatement`
dfp_getWorkflowRequestsByStatement

Examples

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getUserTeamAssociationsByStatement(dat)

## End(Not run)
```

---

dfp_getWorkflowRequestsByStatement

WorkflowRequestService

Description

Provides methods to retrieve and perform actions on WorkflowRequest objects

Usage

```r
dfp_getWorkflowRequestsByStatement(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Details

To use this service, you need to have the new sales management solution enabled on your network. If you do not see a "Sales" tab in <a href="https://www.google.com/dfp">DoubleClick for Publishers (DFP)</a>, you will not be able to use this service.

getWorkflowRequestsByStatement

Gets a list of WorkflowRequest objects that satisfy the given Statement query. The following fields are supported for filtering:

- `id`
- `workflowRuleName`
- `entityType`
- `entityId`
- `approvalStatus`
- `conditionStatus`
- `type`
`dfp_hasCustomPacingCurve`

**Value**

A data frame or list containing all the elements of a getWorkflowRequestsByStatementResponse.

**See Also**

Google Documentation for `getWorkflowRequestsByStatement`.

**Examples**

```r
## Not run:
dat <- list(filterStatement=list('query'="WHERE status='ACTIVE'"))
res <- dfp_getWorkflowRequestsByStatement(dat)

## End(Not run)
```

---

`dfp_hasCustomPacingCurve`

**Description**

Returns whether a custom pacing curve has been uploaded to Google Cloud Storage for a line item. @param lineItemId the ID of the line item

**Usage**

```r
dfp_hasCustomPacingCurve(as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame.
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML.

**Value**

A data frame or list containing all the elements of a hasCustomPacingCurveResponse.

**See Also**

Google Documentation for `hasCustomPacingCurve`.

**Examples**

```r
## Not run:
res <- dfp_hasCustomPacingCurve()

## End(Not run)
```
Description

Creates a new blank network for testing purposes using the current login. Each login (i.e. email address) can only have one test network. Data from any of your existing networks will not be transferred to the new test network. Once the test network is created, the test network can be used in the API by supplying the Network networkCode in the SOAP header or by logging into the Ad Manager UI. Test networks are limited in the following ways:

- Test networks cannot serve ads.
- Because test networks cannot serve ads, reports will always come back without data.
- Since forecasting requires serving history, forecast service results will be faked. See Forecast-Service for more info.
- Test networks are, by default, Ad Manager networks and don’t have any features from Ad Manager 360. To have additional features turned on, please contact your account manager.
- Test networks are limited to 10,000 objects per entity type.

Creates a new blank network for testing purposes using the current login. Each login (i.e. email address) can only have one test network. Data from any of your existing networks will not be transferred to the new test network. Once the test network is created, the test network can be used in the API by supplying the Network networkCode in the SOAP header or by logging into the Ad Manager UI. Test networks are limited in the following ways:

- Test networks cannot serve ads.
- Because test networks cannot serve ads, reports will always come back without data.
- Since forecasting requires serving history, forecast service results will be faked. See Forecast-Service for more info.
- Test networks are, by default, Ad Manager networks and don’t have any features from Ad Manager 360. To have additional features turned on, please contact your account manager.
- Test networks are limited to 10,000 objects per entity type.

Creates a new blank network for testing purposes using the current login. Each login (i.e. email address) can only have one test network. Data from any of your existing networks will not be transferred to the new test network. Once the test network is created, the test network can be used in the API by supplying the Network networkCode in the SOAP header or by logging into the Ad Manager UI. Test networks are limited in the following ways:

- Test networks cannot serve ads.
- Because test networks cannot serve ads, reports will always come back without data.
- Since forecasting requires serving history, forecast service results will be faked. See Forecast-Service for more info.
- Test networks are, by default, Ad Manager networks and don’t have any features from Ad Manager 360. To have additional features turned on, please contact your account manager.
- Test networks are limited to 10,000 objects per entity type.
Usage

dfp_makeTestNetwork(as_df = TRUE, verbose = FALSE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>as_df</td>
<td>a boolean indicating whether to attempt to parse the result into a data.frame</td>
</tr>
<tr>
<td>verbose</td>
<td>a boolean indicating whether to print the service URL and POSTed XML</td>
</tr>
</tbody>
</table>

Value

a data.frame or list containing all the elements of a makeTestNetworkResponse

See Also

Google Documentation for makeTestNetwork

Examples

```r
## Not run:
res <- dfp_makeTestNetwork()
## End(Not run)
```

---

dfp_performAdExclusionRuleAction

**performAdExclusionRuleAction**

Description

Performs action on AdExclusionRule objects that satisfy the given Statement query.

Usage

dfp_performAdExclusionRuleAction(request_data, as_df = TRUE, verbose = FALSE)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_data</td>
<td>a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)</td>
</tr>
<tr>
<td>as_df</td>
<td>a boolean indicating whether to attempt to parse the result into a data.frame</td>
</tr>
<tr>
<td>verbose</td>
<td>a boolean indicating whether to print the service URL and POSTed XML</td>
</tr>
</tbody>
</table>

Value

a data.frame or list containing all the elements of a performAdExclusionRuleActionResponse
dfp_performAdRuleAction

See Also

Google Documentation for performAdExclusionRuleAction

Examples

```r
## Not run:
res <- dfp_performAdExclusionRuleAction(request_data)
## End(Not run)
```

## dfp_performAdRuleAction

`performAdRuleAction`

### Description

Performs actions on AdRule objects that match the given Statement query.

### Usage

```r
dfp_performAdRuleAction(request_data, as_df = TRUE, verbose = FALSE)
```

### Arguments

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

### Value

A `data.frame` or `list` containing all the elements of a `performAdRuleActionResponse`

### See Also

Google Documentation for `performAdRuleAction`

### Examples

```r
## Not run:
res <- dfp_performAdRuleAction(request_data)
## End(Not run)```
**dfp_performAdUnitAction**

**Description**
Performs actions on AdUnit objects that match the given Statement query.

**Usage**
```
dfp_performAdUnitAction(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**
- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**
a data.frame or list containing all the elements of a performAdUnitActionResponse

**See Also**
Google Documentation for `performAdUnitAction`

**Examples**
```
## Not run:
res <- dfp_performAdUnitAction(request_data)

## End(Not run)
```

**dfp_performAudienceSegmentAction**

**Description**
Performs the given AudienceSegmentAction on the set of segments identified by the given statement.
dfp_performBaseRateAction

Usage

\[
dfp\_perform\_Base\_Rate\_Action(request\_data, \text{as}\_df = \text{TRUE}, \\
\quad \text{verbose} = \text{FALSE})
\]

Arguments

- **request\_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as\_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a performAudienceSegmentActionResponse

See Also

Google Documentation for performAudienceSegmentAction

Examples

```r
## Not run:
res <- dfp_performAudienceSegmentAction(request_data)
## End(Not run)
```

---

dfp_performBaseRateAction

**performBaseRateAction**

---

Description

Performs actions on BaseRate objects that satisfy the given Statement query.

Usage

\[
dfp\_perform\_Base\_Rate\_Action(request\_data, \text{as}\_df = \text{TRUE}, \text{verbose} = \text{FALSE})
\]

Arguments

- **request\_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as\_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML
Value

a data.frame or list containing all the elements of a performBaseRateActionResponse

See Also

Google Documentation for performBaseRateAction

Examples

```r
## Not run:
res <- dfp_performBaseRateAction(request_data)
## End(Not run)
```

---

### dfp_performCdnConfigurationAction

#### performCdnConfigurationAction

**Description**

Performs actions on CdnConfiguration objects that match the given [@link Statement query].

**Usage**

```r
dfp_performCdnConfigurationAction(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a performCdnConfigurationActionResponse

**See Also**

Google Documentation for performCdnConfigurationAction

**Examples**

```r
## Not run:
res <- dfp_performCdnConfigurationAction(request_data)
## End(Not run)
```
dfp_performContentBundleAction

**Description**

Performs actions on ContentBundle objects that match the given Statement query.

**Usage**

```r
dfp_performContentBundleAction(request_data, as_df = TRUE,
                               verbose = FALSE)
```

**Arguments**

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

A `data.frame` or list containing all the elements of a `performContentBundleActionResponse`

**See Also**

[Google Documentation for performContentBundleAction](#)

**Examples**

```r
## Not run:
res <- dfp_performContentBundleAction(request_data)

## End(Not run)
```

dfp_performCreativeWrapperAction

**Description**

Performs actions on CreativeWrapper objects that match the given Statement query.
Usage

dfp_performCreativeWrapperAction(request_data, as_df = TRUE,
   verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request
   (XML format, but passed as character string)

as_df         a boolean indicating whether to attempt to parse the result into a data.frame

verbose       a boolean indicating whether to print the service URL and POSTed XML

Value

   a data.frame or list containing all the elements of a performCreativeWrapperActionResponse

See Also

   Google Documentation for performCreativeWrapperAction

Examples

   ## Not run:
   res <- dfp_performCreativeWrapperAction(request_data)

   ## End(Not run)

---

dfp_performCustomFieldAction

   performCustomFieldAction

Description

   Performs actions on CustomField objects that match the given Statement query.

Usage

   dfp_performCustomFieldAction(request_data, as_df = TRUE,
      verbose = FALSE)

Arguments

request_data   a list or data.frame of data elements to be formatted for a SOAP request
   (XML format, but passed as character string)

as_df          a boolean indicating whether to attempt to parse the result into a data.frame

verbose        a boolean indicating whether to print the service URL and POSTed XML
**dfp_performCustomTargetingKeyAction**

**Value**

a data.frame or list containing all the elements of a performCustomFieldActionResponse

**See Also**

Google Documentation for performCustomFieldAction

**Examples**

```r
## Not run:
res <- dfp_performCustomFieldAction(request_data)
```

## End(Not run)

**dfp_performCustomTargetingKeyAction**

**performCustomTargetingKeyAction**

**Description**

Performs actions on CustomTargetingKey objects that match the given Statement query.

**Usage**

```r
dfp_performCustomTargetingKeyAction(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data` a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a performCustomTargetingKeyActionResponse

**See Also**

Google Documentation for performCustomTargetingKeyAction

**Examples**

```r
## Not run:
res <- dfp_performCustomTargetingKeyAction(request_data)
```

## End(Not run)
dfp_performCustomTargetingValueAction

Description

Performs actions on CustomTargetingValue objects that match the given Statement query.

Usage

dfp_performCustomTargetingValueAction(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df         a boolean indicating whether to attempt to parse the result into a data.frame
verbose       a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a performCustomTargetingValueActionResponse

See Also

Google Documentation for performCustomTargetingValueAction

Examples

## Not run:
res <- dfp_performCustomTargetingValueAction(request_data)

## End(Not run)


dfp_performDaiAuthenticationKeyAction

Description

Performs actions on DaiAuthenticationKey objects that match the given Statement query. DAI authentication keys cannot be deactivated if there are active LiveStreamEvents or Content Sources that are using them.
Usage

dfp_performDaiAuthenticationKeyAction(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a performDaiAuthenticationKeyActionResponse

See Also

Google Documentation for performDaiAuthenticationKeyAction

Examples

## Not run:
res <- dfp_performDaiAuthenticationKeyAction(request_data)

## End(Not run)

dfp_performExchangeRateAction

performExchangeRateAction

Description

Performs an action on ExchangeRate objects that satisfy the given Statement query. The following fields are supported for filtering:

- id
- currencyCode
- refreshRate
- direction
- exchangeRate

Usage

dfp_performExchangeRateAction(request_data, as_df = TRUE, verbose = FALSE)
**dfp_performLabelAction**

**Arguments**
- **request_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML

**Value**
- a data.frame or list containing all the elements of a performExchangeRateActionResponse

**See Also**
- Google Documentation for performExchangeRateAction

**Examples**

```r
## Not run:
res <- dfp_performExchangeRateAction(request_data)
## End(Not run)
```

---

**dfp_performLabelAction**

**performLabelAction**

**Description**
Performs actions on Label objects that match the given Statement query.

**Usage**

```r
dfp_performLabelAction(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**
- **request_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML

**Value**
- a data.frame or list containing all the elements of a performLabelActionResponse

**See Also**
- Google Documentation for performLabelAction
dfp_performLineItemAction

**Examples**

```r
## Not run:
res <- dfp_performLineItemAction(request_data)

## End(Not run)
```

dfp_performLineItemAction

### Description

Performs actions on LineItem objects that match the given Statement query.

### Usage

`dfp_performLineItemAction(request_data, as_df = TRUE, verbose = FALSE)`

### Arguments

- **request_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML

### Value

A data.frame or list containing all the elements of a performLineItemActionResponse

### See Also

[Google Documentation for performLineItemAction](#)

### Examples

```r
## Not run:
res <- dfp_performLineItemAction(request_data)

## End(Not run)
```
**dfp_performLineItemCreativeAssociationAction**

**performLineItemCreativeAssociationAction**

**Description**

Performs actions on LineItemCreativeAssociation objects that match the given Statement query.

**Usage**

```r
dfp_performLineItemCreativeAssociationAction(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

A data.frame or list containing all the elements of a `performLineItemCreativeAssociationActionResponse`

**See Also**

[Google Documentation for performLineItemCreativeAssociationAction](#)

**Examples**

```r
## Not run:
res <- dfp_performLineItemCreativeAssociationAction(request_data)

## End(Not run)
```

---

**dfp_performLiveStreamEventAction**

**performLiveStreamEventAction**

**Description**

Performs actions on LiveStreamEvent objects that match the given Statement query.
dfp_performMobileApplicationAction

Usage

dfp_performLiveStreamEventAction(request_data, as_df = TRUE,
   verbose = FALSE)

Arguments

  request_data  a list or data.frame of data elements to be formatted for a SOAP request
                 (XML format, but passed as character string)
  as_df         a boolean indicating whether to attempt to parse the result into a data.frame
  verbose       a boolean indicating whether to print the service URL and POSTed XML

Value

  a data.frame or list containing all the elements of a performLiveStreamEventActionResponse

See Also

  Google Documentation for performLiveStreamEventAction

Examples

## Not run:
res <- dfp_performLiveStreamEventAction(request_data)

## End(Not run)

---

dfp_performMobileApplicationAction

        performMobileApplicationAction

Description

Performs an action on MobileApplication mobile applications.

Usage


dfp_performMobileApplicationAction(request_data, as_df = TRUE,
   verbose = FALSE)

Arguments

  request_data  a list or data.frame of data elements to be formatted for a SOAP request
                 (XML format, but passed as character string)
  as_df         a boolean indicating whether to attempt to parse the result into a data.frame
  verbose       a boolean indicating whether to print the service URL and POSTed XML
Value

A `data.frame` or `list` containing all the elements of a `performMobileApplicationActionResponse`.

See Also

Google Documentation for `performMobileApplicationAction`.

Examples

```r
## Not run:
res <- dfp_performMobileApplicationAction(request_data)
## End(Not run)
```

---

**dfp_performNativeStyleAction**

### Description

Performs actions on NativeStyle native styles that match the given Statement.

### Usage

```r
dfp_performNativeStyleAction(request_data, as_df = TRUE, verbose = FALSE)
```

### Arguments

- `request_data`: A `list` or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string).
- `as_df`: A boolean indicating whether to attempt to parse the result into a `data.frame`.
- `verbose`: A boolean indicating whether to print the service URL and POSTed XML.

### Value

A `data.frame` or `list` containing all the elements of a `performNativeStyleActionResponse`.

See Also

Google Documentation for `performNativeStyleAction`.

Examples

```r
## Not run:
res <- dfp_performNativeStyleAction(request_data)
## End(Not run)
```
dfp_performOrderAction

**Description**

Performs actions on Order objects that match the given Statement query.

**Usage**

```r
dfp_performOrderAction(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

A data frame or list containing all the elements of a `performOrderActionResponse`

**See Also**

- [Google Documentation for performOrderAction](#)

**Examples**

```r
## Not run:
res <- dfp_performOrderAction(request_data)
## End(Not run)
```

dfp_performPackageAction

**Description**

Performs actions on Package objects that match the given Statement.

**Usage**

```r
dfp_performPackageAction(request_data, as_df = TRUE, verbose = FALSE)
```
dfp_performPlacementAction

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

A data.frame or list containing all the elements of a performPackageActionResponse

See Also

Google Documentation for performPackageAction

Examples

```r
## Not run:
res <- dfp_performPackageAction(request_data)
## End(Not run)
```

dfp_performPlacementAction

performPlacementAction

Description

Performs actions on Placement objects that match the given Statement query.

Usage

```r
dfp_performPlacementAction(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

A data.frame or list containing all the elements of a performPlacementActionResponse

See Also

Google Documentation for performPlacementAction
**dfp_performProductAction**

### Description
Performs action on Product objects that satisfy the given Statement.

### Usage
```
dfp_performProductAction(request_data, as_df = TRUE, verbose = FALSE)
```

### Arguments
- **request_data**
  - a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**
  - a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**
  - a boolean indicating whether to print the service URL and POSTed XML

### Value
a data.frame or list containing all the elements of a performProductActionResponse

### See Also
[Google Documentation for performProductAction](#)

### Examples
```
## Not run:
res <- dfp_performProductAction(request_data)
## End(Not run)
```
**Description**

Performs actions on ProductPackage objects that match the given `Statement` query.

**Usage**

```r
dfp_performProductPackageAction(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data` a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Value**

A `data.frame` or list containing all the elements of a `performProductPackageActionResponse`

**See Also**

[Google Documentation for performProductPackageAction](#)

**Examples**

```r
## Not run:
res <- dfp_performProductPackageAction(request_data)
## End(Not run)
```

---

**dfp_performProductPackageItemAction**

**performProductPackageItemAction**

**Description**

Performs actions on ProductPackageItem objects that satisfy the given `Statement` query.
Usage

dfp_performProductPackageItemAction(request_data, as_df = TRUE, 
   verbose = FALSE)

Arguments

  request_data a list or data.frame of data elements to be formatted for a SOAP request 
                 (XML format, but passed as character string)

  as_df a boolean indicating whether to attempt to parse the result into a data.frame

  verbose a boolean indicating whether to print the service URL and POSTed XML

Value

  a data.frame or list containing all the elements of a performProductPackageItemActionResponse

See Also

Google Documentation for performProductPackageItemAction

Examples

## Not run:
res <- dfp_performProductPackageItemAction(request_data)

## End(Not run)
dfp_performProposalAction

Value

a data.frame or list containing all the elements of a performProductTemplateActionResponse

See Also

Google Documentation for performProductTemplateAction

Examples

```r
## Not run:
res <- dfp_performProductTemplateAction(request_data)
## End(Not run)
```

```r
dfp_performProposalAction
```

Description

Performs actions on Proposal objects that match the given Statement query. The following fields are also required when submitting proposals for approval:

- Proposal advertiser
- Proposal primarySalesperson
- Proposal primaryTraffickerId

Usage

dfp_performProposalAction(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a performProposalActionResponse

See Also

Google Documentation for performProposalAction
dfp_performProposalLineItemAction

Examples

```r
## Not run:
res <- dfp_performProposalAction(request_data)

## End(Not run)
```

dfp_performProposalLineItemAction

*performProposalLineItemAction*

Description

Performs actions on ProposalLineItem objects that match the given Statement query.

Usage

```r
dfp_performProposalLineItemAction(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a performProposalLineItemActionResponse

See Also

Google Documentation for performProposalLineItemAction

Examples

```r
## Not run:
res <- dfp_performProposalLineItemAction(request_data)

## End(Not run)
```
dfp_performRateCardAction

*performRateCardAction*

**Description**

Performs action on RateCard objects that satisfy the given Statement query.

**Usage**

```r
dfp_performRateCardAction(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- **request_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML

**Value**

A data.frame or list containing all the elements of a performRateCardActionResponse

**See Also**

[Google Documentation for performRateCardAction](#)

**Examples**

```r
## Not run:
res <- dfp_performRateCardAction(request_data)
## End(Not run)
```

dfp_performReconciliationOrderReportAction

*performReconciliationOrderReportAction*

**Description**

Performs actions on the ReconciliationOrderReport objects that match the given Statement query. The following fields are supported for filtering:

- orderId
- proposalId
- reconciliationReportId
Usage

dfp_performReconciliationOrderReportAction(request_data, as_df = TRUE, 
   verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request 
   (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame 
   verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a performReconciliationOrderReportAction-
   Response

See Also

Google Documentation for performReconciliationOrderReportAction

Examples

## Not run:
res <- dfp_performReconciliationOrderReportAction(request_data)

## End(Not run)

Description

Performs actions on SuggestedAdUnit objects that match the given Statement query. The following 
fields are supported for filtering:

- id
- numRequests

Usage

dfp_performSuggestedAdUnitAction(request_data, as_df = TRUE, 
   verbose = FALSE)
dfp_performTeamAction

**Arguments**

- `request_data` a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a performSuggestedAdUnitActionResponse

**See Also**

Google Documentation for performSuggestedAdUnitAction

**Examples**

```r
## Not run:
res <- dfp_performSuggestedAdUnitAction(request_data)

## End(Not run)
```

---

dfp_performTeamAction  performTeamAction

**Description**

Performs actions on Team objects that match the given Statement query.

**Usage**

```r
dfp_performTeamAction(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data` a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a performTeamActionResponse

**See Also**

Google Documentation for performTeamAction
dfp_performUserAction

Examples

```r
## Not run:
res <- dfp_performTeamAction(request_data)

## End(Not run)
```

---

**dfp_performUserAction performUserAction**

**Description**

Performs actions on User objects that match the given Statement query.

**Usage**

```r
dfp_performUserAction(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data` a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Value**

A `data.frame` or `list` containing all the elements of a `performUserActionResponse`

**See Also**

[Google Documentation for performUserAction](#)

**Examples**

```r
## Not run:
res <- dfp_performUserAction(request_data)

## End(Not run)
```
dfp_performUserTeamAssociationAction

performUserTeamAssociationAction

Description

Performs actions on UserTeamAssociation objects that match the given Statement query.

Usage

dfp_performUserTeamAssociationAction(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a performUserTeamAssociationActionResponse

See Also

Google Documentation for performUserTeamAssociationAction

Examples

## Not run:
res <- dfp_performUserTeamAssociationAction(request_data)

## End(Not run)

dfp_performWorkflowRequestAction

performWorkflowRequestAction

Description

Perform actions on WorkflowRequest objects that match the given Statement query.
**dfp_registerSessionsForMonitoring**

**Usage**

```r
dfp_registerSessionsForMonitoring(as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `as_df` a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Description**

Registers the specified list of session IDs for monitoring. Once the session IDs have been registered, all logged information about the sessions will be persisted and can be viewed via the Ad Manager UI. A session ID is a unique identifier of a single user watching a live stream event.

**Usage**

```r
dfp_registerSessionsForMonitoring(as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `as_df` a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose` a boolean indicating whether to print the service URL and POSTed XML
Value

a data.frame or list containing all the elements of a registerSessionsForMonitoringResponse

See Also

Google Documentation for registerSessionsForMonitoring

Examples

```r
## Not run:
res <- dfp_registerSessionsForMonitoring()

## End(Not run)
```

dfp_report_url_to_dataframe

**Take report URL and convert to data.frame**

Description

Receive a URL (usually from the ReportService) and download data from that URL. Currently, the exportFormat must have been set for CSV_DUMP

Usage

dfp_report_url_to_dataframe(report_url, exportFormat='CSV_DUMP')

Arguments

- `report_url`: a URL character string returned from the function `dfp_getReportDownloadURL`
- `exportFormat`: a character string naming what type of exportFormat was provided to `dfp_getReportDownloadURL`. This is used to determine how to parse the results.

Value

a data.frame of report results from the specified URL
dfp_runReportJob

Description

Initiates the execution of a ReportQuery on the server. The following fields are required:

- ReportJob reportQuery

Usage

dfp_runReportJob(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

- a data.frame or list containing all the elements of a runReportJobResponse

See Also

Google Documentation for runReportJob

Examples

```r
## Not run:
request_data <- list(reportJob=
  list(reportQuery=
    list(dimensions='MONTH_AND_YEAR',
      dimensions='AD_UNIT_ID',
      adUnitView='FLAT',
      columns='AD_SERVER_CLICKS',
      dateRangeType='LAST_WEEK'))

# the result is a list and most importantly the ID is included for checking its status
dfp_runReportJob_result <- dfp_runReportJob(request_data)
dfp_runReportJob_result$id

## End(Not run)
```
Description

Provides methods for executing a PQL Statement to retrieve information from the system. In order to support the selection of columns of interest from various tables, Statement objects support a "select" clause. An example query text might be "select CountryCode, Name from Geo_Target", where CountryCode and Name are columns of interest and Geo_Target is the table.

Usage

dfp_select(request_data, verbose = FALSE)

Arguments

request_data | a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
verbose | a boolean indicating whether to print the service URL and POSTed XML

Details

The following tables are supported:

- Geo_Target
- Bandwidth_Group
- Browser
- Browser_Language
- Device_Capability
- Device_Category
- Device_Manufacturer
- Mobile_Carrier
- Mobile_Device
- Mobile_Device_Submodel
- Operating_System
- Operating_System_Version
- Third_Party_Company
- Line_Item
- Ad_Unit
- User
- Exchange_Rate
- Programmatic_Buyer
- Audience_Segment_Category
- Audience_Segment
- Proposal_Retraction_Reason
- Time_Zone
- Proposal_Terms_And_Conditions
- Change_History
- ad_category

Visit the See Also section below to proceed to Google and view columns in each of these tables.

select

retrieves rows of data that satisfy the given Statement query from the system.

Value

a data.frame or list containing all the elements of a selectResponse

See Also

Google Documentation for select

Examples

```r
## Not run:
request_data <- list(selectStatement=
  list(query='SELECT Id, Name, Targeting FROM LineItem LIMIT 3'))
dfp_select_result <- dfp_select(request_data)

request_data <- list(selectStatement=
  list(query="SELECT Id, Name, CanonicalParentId, CountryCode, Type
        FROM Geo_Target
        WHERE CountryCode='US' AND (TYPE='STATE' OR TYPE='COUNTY')")
us_geos <- dfp_select(request_data)

## End(Not run)
```

dfp_updateActivities  updateActivities

Description

Updates the specified Activity objects.
**dfp_updateActivityGroups**

**Usage**

dfp_updateActivities(request_data, as_df = TRUE, verbose = FALSE)

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a updateActivitiesResponse

**See Also**

Google Documentation for updateActivities

**Examples**

```r
## Not run:
res <- dfp_updateActivities(request_data)

## End(Not run)
```

---

**dfp_updateActivityGroups**

### updateActivityGroups

**Description**

Updates the specified ActivityGroup objects.

**Usage**

dfp_updateActivityGroups(request_data, as_df = TRUE, verbose = FALSE)

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a updateActivityGroupsResponse
dfp_updateAdExclusionRules

See Also

Google Documentation for updateActivityGroups

Examples

```r
## Not run:
res <- dfp_updateActivityGroups(request_data)
## End(Not run)
```

```r
dfp_updateAdExclusionRules

updateAdExclusionRules
```

Description

Updates the specified AdExclusionRule objects.

Usage

```r
dfp_updateAdExclusionRules(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Value

A `data.frame` or list containing all the elements of a `updateAdExclusionRulesResponse`

See Also

Google Documentation for `updateAdExclusionRules`

Examples

```r
## Not run:
res <- dfp_updateAdExclusionRules(request_data)
## End(Not run)
```
dfp_updateAdRules  updateAdRules

Description

Updates the specified AdRule objects.

Usage

dfp_updateAdRules(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateAdRulesResponse

See Also

Google Documentation for updateAdRules

Examples

```r
## Not run:
res <- dfp_updateAdRules(request_data)

## End(Not run)
```

dfp_updateAdUnits  updateAdUnits

Description

Updates the specified AdUnit objects.

Usage

dfp_updateAdUnits(request_data, as_df = TRUE, verbose = FALSE)
**Arguments**

- **request_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a updateAudienceSegmentsResponse

**See Also**

Google Documentation for updateAudienceSegments

**Examples**

```r
## Not run:
res <- dfp_updateAudienceSegments(request_data)
## End(Not run)
```

---

**Description**

Updates the given RuleBasedFirstPartyAudienceSegment objects.

**Usage**

dfp_updateAudienceSegments(request_data, as_df = TRUE, verbose = FALSE)

**Arguments**

- **request_data**: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- **as_df**: a boolean indicating whether to attempt to parse the result into a data.frame
- **verbose**: a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a updateAudienceSegmentsResponse

**See Also**

Google Documentation for updateAudienceSegments
Examples

```r
## Not run:
res <- dfp_updateBaseRates(request_data)

## End(Not run)
```

---

dfp_updateBaseRates  updateBaseRates

**Description**

Updates the specified BaseRate objects.

**Usage**

```r
dfp_updateBaseRates(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>request_data</td>
<td>a list or data.frame of data elements to be formatted for a SOAP request</td>
</tr>
<tr>
<td></td>
<td>(XML format, but passed as character string)</td>
</tr>
<tr>
<td>as_df</td>
<td>a boolean indicating whether to attempt to parse the result into a data.frame</td>
</tr>
<tr>
<td>verbose</td>
<td>a boolean indicating whether to print the service URL and POSTed XML</td>
</tr>
</tbody>
</table>

**Value**

a data.frame or list containing all the elements of a updateBaseRatesResponse

**See Also**

[Google Documentation for updateBaseRates](#)

**Examples**

```r
## Not run:
res <- dfp_updateBaseRates(request_data)

## End(Not run)
```
dfp_updateCdnConfigurations

Description

Updates the specified CdnConfiguration objects.

Usage

dfp_updateCdnConfigurations(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateCdnConfigurationsResponse

See Also

Google Documentation for updateCdnConfigurations

Examples

```r
## Not run:
res <- dfp_updateCdnConfigurations(request_data)
## End(Not run)
```

dfp_updateCompanies

Description

Updates the specified Company objects.

Usage

dfp_updateCompanies(request_data, as_df = TRUE, verbose = FALSE)
Arguments
request_data   a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df          a boolean indicating whether to attempt to parse the result into a data.frame
verbose        a boolean indicating whether to print the service URL and POSTed XML

Value
a data.frame or list containing all the elements of a updateCompaniesResponse

See Also
Google Documentation for updateCompanies

Examples
## Not run:
res <- dfp_updateCompanies(request_data)
## End(Not run)

dfp_updateContacts  updateContacts

Description
Updates the specified Contact objects.

Usage
dfp_updateContacts(request_data, as_df = TRUE, verbose = FALSE)

Arguments
request_data   a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df          a boolean indicating whether to attempt to parse the result into a data.frame
verbose        a boolean indicating whether to print the service URL and POSTed XML

Value
a data.frame or list containing all the elements of a updateContactsResponse

See Also
Google Documentation for updateContacts
**dfp_updateContentBundles**

**updateContentBundles**

**Description**

Updates the specified ContentBundle objects.

**Usage**

```r
dfp_updateContentBundles(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

A `data.frame` or list containing all the elements of a `updateContentBundlesResponse`

**See Also**

[Google Documentation for updateContentBundles](#)

**Examples**

```r
## Not run:
res <- dfp_updateContacts(request_data)

## End(Not run)
```
dfp_updateCreatives   updateCreatives

Description
Updates the specified Creative objects.

Usage
dfp_updateCreatives(request_data, as_df = TRUE, verbose = FALSE)

Arguments
request_data       a list or data.frame of data elements to be formatted for a SOAP request
                   (XML format, but passed as character string)
as_df              a boolean indicating whether to attempt to parse the result into a data.frame
verbose            a boolean indicating whether to print the service URL and POSTed XML

Value
a data.frame or list containing all the elements of a updateCreativesResponse

See Also
Google Documentation for updateCreatives

Examples
## Not run:
res <- dfp_updateCreatives(request_data)
## End(Not run)

dfp_updateCreativeSet   updateCreativeSet

Description
Updates the specified CreativeSet.

Usage
dfp_updateCreativeSet(request_data, as_df = TRUE, verbose = FALSE)
**dfp_updateCreativeWrappers**

**Arguments**

- `request_data` a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Value**

- a data.frame or list containing all the elements of a updateCreativeSetResponse

**See Also**

Google Documentation for updateCreativeSet

**Examples**

```r
## Not run:
res <- dfp_updateCreativeSet(request_data)
## End(Not run)
```

---

**dfp_updateCreativeWrappers**

**updateCreativeWrappers**

**Description**

Updates the specified CreativeWrapper objects.

**Usage**

```
dfp_updateCreativeWrappers(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data` a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df` a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose` a boolean indicating whether to print the service URL and POSTed XML

**Value**

- a data.frame or list containing all the elements of a updateCreativeWrappersResponse

**See Also**

Google Documentation for updateCreativeWrappers
dfp_updateCustomFieldOptions

updateCustomFieldOptions

Description

Updates the specified CustomFieldOption objects.

Usage

dfp_updateCustomFieldOptions(request_data, as_df = TRUE,
  verbose = FALSE)

Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request
  (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateCustomFieldOptionsResponse

See Also

Google Documentation for updateCustomFieldOptions

Examples

```r
## Not run:
res <- dfp_updateCustomFieldOptions(request_data)
## End(Not run)
```
**dfp_updateCustomFields**

**updateCustomFields**

**Description**

Updates the specified CustomField objects.

**Usage**

```
dfp_updateCustomFields(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

A data.frame or list containing all the elements of a `updateCustomFieldsResponse`

**See Also**

Google Documentation for `updateCustomFields`

**Examples**

```r
## Not run:
res <- dfp_updateCustomFields(request_data)
## End(Not run)
```

---

**dfp_updateCustomTargetingKeys**

**updateCustomTargetingKeys**

**Description**

Updates the specified CustomTargetingKey objects.

**Usage**

```
dfp_updateCustomTargetingKeys(request_data, as_df = TRUE, verbose = FALSE)
```
dfp_updateCustomTargetingValues

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df  a boolean indicating whether to attempt to parse the result into a data.frame

verbose  a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateCustomTargetingKeysResponse

See Also

Google Documentation for updateCustomTargetingKeys

Examples

```r
## Not run:
res <- dfp_updateCustomTargetingKeys(request_data)

## End(Not run)
```

dfp_updateCustomTargetingValues

updateCustomTargetingValues

Description

Updates the specified CustomTargetingValue objects.

Usage

```r
dfp_updateCustomTargetingValues(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df  a boolean indicating whether to attempt to parse the result into a data.frame

verbose  a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateCustomTargetingValuesResponse
dfp_updateDaiAuthenticationKeys

See Also

Google Documentation for updateCustomTargetingValues

Examples

```r
## Not run:
res <- dfp_updateCustomTargetingValues(request_data)

## End(Not run)
```

dfp_updateDaiAuthenticationKeys

Description

Updates the specified DaiAuthenticationKey objects.

Usage

```r
dfp_updateDaiAuthenticationKeys(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Value

A `data.frame` or list containing all the elements of a `updateDaiAuthenticationKeysResponse`

See Also

Google Documentation for `updateDaiAuthenticationKeys`

Examples

```r
## Not run:
res <- dfp_updateDaiAuthenticationKeys(request_data)

## End(Not run)
```
dfp_updateExchangeRates

updateExchangeRates

Description

Updates the specified ExchangeRate objects.

Usage

dfp_updateExchangeRates(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateExchangeRatesResponse

See Also

Google Documentation for updateExchangeRates

Examples

## Not run:
res <- dfp_updateExchangeRates(request_data)

## End(Not run)

dfp_updateLabels

updateLabels

Description

Updates the specified Label objects.

Usage

dfp_updateLabels(request_data, as_df = TRUE, verbose = FALSE)
dfp_updateLineItemCreativeAssociations

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

A data.frame or list containing all the elements of a updateLineItemCreativeAssociationsResponse

See Also

Google Documentation for updateLabels

Examples

```r
## Not run:
res <- dfp_updateLabels(request_data)
## End(Not run)
```

updateLineItemCreativeAssociations

Description

Updates the specified LineItemCreativeAssociation objects

Usage

```r
dfp_updateLineItemCreativeAssociations(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

A data.frame or list containing all the elements of a updateLineItemCreativeAssociationsResponse
See Also

Google Documentation for updateLineItemCreativeAssociations

Examples

```
## Not run:
res <- dfp_updateLineItemCreativeAssociations(request_data)
## End(Not run)
```

---

dfp_updateLineItems   updateLineItems

Description

Updates the specified LineItem objects.

Usage

```
dfp_updateLineItems(request_data, as_df = TRUE, verbose = FALSE)
```

Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateLineItemsResponse

See Also

Google Documentation for updateLineItems

Examples

```
## Not run:
res <- dfp_updateLineItems(request_data)
## End(Not run)
```
dfp_updateLiveStreamEvents

updateLiveStreamEvents

Description

Updates the specified LiveStreamEvent objects.

Usage

dfp_updateLiveStreamEvents(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df          a boolean indicating whether to attempt to parse the result into a data.frame
verbose        a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateLiveStreamEventsResponse

See Also

Google Documentation for updateLiveStreamEvents

Examples

## Not run:
res <- dfp_updateLiveStreamEvents(request_data)

## End(Not run)

dfp_updateMobileApplications

updateMobileApplications

Description

Updates the specified MobileApplication mobile applications.

Usage

dfp_updateMobileApplications(request_data, as_df = TRUE, verbose = FALSE)
dfp_updateNativeStyles

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df a boolean indicating whether to attempt to parse the result into a data.frame

verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateMobileApplicationsResponse

See Also

Google Documentation for updateMobileApplications

Examples

```r
## Not run:
res <- dfp_updateMobileApplications(request_data)

## End(Not run)
```

dfp_updateNativeStyles

**updateNativeStyles**

Description

Updates the specified NativeStyle objects.

Usage

dfp_updateNativeStyles(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df a boolean indicating whether to attempt to parse the result into a data.frame

verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateNativeStylesResponse

See Also

Google Documentation for updateNativeStyles
**dfp_updateNetwork**

### Examples

```r
## Not run:
res <- dfp_updateNativeStyles(request_data)

## End(Not run)
```

---

**Description**

Updates the specified network. Currently, only the network display name can be updated.

**Usage**

```
dfp_updateNetwork(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

A `data.frame` or `list` containing all the elements of a `updateNetworkResponse`

**See Also**

[Google Documentation for updateNetwork](#)

**Examples**

```r
## Not run:
res <- dfp_updateNetwork(request_data)

## End(Not run)
```
dfp_updateOrders  

*bupdateOrders*

**Description**

Updates the specified Order objects.

**Usage**

```r
dfp_updateOrders(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a updateOrdersResponse

**See Also**

Google Documentation for updateOrders

**Examples**

```r
## Not run:
res <- dfp_updateOrders(request_data)

## End(Not run)
```

dfp_updatePackages  

*bupdatePackages*

**Description**

Updates the specified Package objects.

**Usage**

```r
dfp_updatePackages(request_data, as_df = TRUE, verbose = FALSE)
```
dfp_updatePlacements

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df  a boolean indicating whether to attempt to parse the result into a data.frame

verbose  a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updatePackagesResponse

See Also

Google Documentation for updatePackages

Examples

CC not runZ
res <- dfp_updatePackages(request_data)
CC endHnot runI

_________________________________________________________________________

dfp_updatePlacements  updatePlacements

Description

Updates the specified Placement objects.

Usage

dfp_updatePlacements(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df  a boolean indicating whether to attempt to parse the result into a data.frame

verbose  a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updatePlacementsResponse

See Also

Google Documentation for updatePlacements
**Examples**

```r
## Not run:
res <- dfp_updatePremiumRates(request_data)

## End(Not run)
```

---

**dfp_updatePremiumRates**

*updatePremiumRates*

---

**Description**

Updates the specified PremiumRate objects.

**Usage**

```r
dfp_updatePremiumRates(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

a data.frame or list containing all the elements of a updatePremiumRatesResponse

**See Also**

- Google Documentation for updatePremiumRates

**Examples**

```r
## Not run:
res <- dfp_updatePremiumRates(request_data)

## End(Not run)
```
dfp_updateProductPackageItems

Description

Updates the specified ProductPackageItem objects.

Usage

dfp_updateProductPackageItems(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateProductPackageItemsResponse

See Also

Google Documentation for updateProductPackageItems

Examples

## Not run:
res <- dfp_updateProductPackageItems(request_data)
## End(Not run)

dfp_updateProductPackages

Description

Updates the specified ProductPackage objects.

Usage

dfp_updateProductPackages(request_data, as_df = TRUE, verbose = FALSE)
dfp_updateProducts

Arguments
request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Value
a data.frame or list containing all the elements of a updateProductPackagesResponse

See Also
Google Documentation for updateProductPackages

Examples
## Not run:
res <- dfp_updateProductPackages(request_data)
## End(Not run)

dfp_updateProducts  updateProducts

Description
Updates the specified Product objects. Note non-updatable fields will not be backfilled.

Usage
dfp_updateProducts(request_data, as_df = TRUE, verbose = FALSE)

Arguments
request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df a boolean indicating whether to attempt to parse the result into a data.frame
verbose a boolean indicating whether to print the service URL and POSTed XML

Value
a data.frame or list containing all the elements of a updateProductsResponse

See Also
Google Documentation for updateProducts
dfp_updateProductTemplates

Description

Updates the specified ProductTemplate objects.

Usage

dfp_updateProductTemplates(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df        a boolean indicating whether to attempt to parse the result into a data.frame

verbose      a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateProductTemplatesResponse

See Also

Google Documentation for updateProductTemplates

Examples

## Not run:
res <- dfp_updateProducts(request_data)

## End(Not run)
dfp_updateProposals

**updateProposals**

**Description**

Updates the specified Proposal objects.

**Usage**

```r
dfp_updateProposals(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

A data.frame or list containing all the elements of a `updateProposalsResponse`

**See Also**

Google Documentation for `updateProposals`

**Examples**

```r
## Not run:
res <- dfp_updateProposals(request_data)
## End(Not run)
```

---

**dfp_updateProposalLineItems**

**updateProposalLineItems**

**Description**

Updates the specified ProposalLineItem objects.

**Usage**

```r
dfp_updateProposalLineItems(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

A data.frame or list containing all the elements of a `updateProposalLineItemsResponse`

**See Also**

Google Documentation for `updateProposalLineItems`

**Examples**

```r
## Not run:
res <- dfp_updateProposalLineItems(request_data)
## End(Not run)
```
### dfp_updateRateCards

**Description**

Updates a list of RateCard objects.

**Usage**

```r
dfp_updateRateCards(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

A `data.frame` or `list` containing all the elements of a `updateRateCardsResponse`

**See Also**

- [Google Documentation for updateRateCards](https://www.criteo.com/developers/docs/)

---

### dfp_updateRateCards

**Arguments**

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

A `data.frame` or `list` containing all the elements of an `updateProposalsResponse`

**See Also**

- [Google Documentation for updateProposals](https://www.criteo.com/developers/docs/)

---

**Examples**

```r
## Not run:
res <- dfp_updateProposals(request_data)

## End(Not run)
```
dfp_updateReconciliationLineItemReports

updateReconciliationLineItemReports

Description

Updates a list of ReconciliationLineItemReport objects which belong to same ReconciliationReport.

Usage

dfp_updateReconciliationLineItemReports(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateReconciliationLineItemReportsResponse

See Also

Google Documentation for updateReconciliationLineItemReports

Examples

```r
## Not run:
res <- dfp_updateRateCards(request_data)

## End(Not run)
```
dfp_updateReconciliationOrderReports

updateReconciliationOrderReports

Description

Updates a list of ReconciliationOrderReport reconciliation order reports which belong to a ReconciliationReport.

Usage

dfp_updateReconciliationOrderReports(request_data, as_df = TRUE, verbose = FALSE)

Arguments

request_data  a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
as_df  a boolean indicating whether to attempt to parse the result into a data.frame
verbose  a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateReconciliationOrderReportsResponse

See Also

Google Documentation for updateReconciliationOrderReports

Examples

## Not run:
res <- dfp_updateReconciliationOrderReports(request_data)

## End(Not run)

dfp_updateReconciliationReportRows

updateReconciliationReportRows

Description

Updates a list of ReconciliationReportRow which belong to same ReconciliationReport.
Usage

dfp_updateReconciliationReportRows(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Value

A data.frame or list containing all the elements of an `updateReconciliationReportRowsResponse`

See Also

- Google Documentation for `updateReconciliationReportRows`

Examples

```r
## Not run:
res <- dfp_updateReconciliationReportRows(request_data)
## End(Not run)
```
dfp_updateTeams

Value

a data.frame or list containing all the elements of a updateReconciliationReportsResponse

See Also

Google Documentation for updateReconciliationReports

Examples

```r
## Not run:
res <- dfp_updateReconciliationReports(request_data)
## End(Not run)
```

dfp_updateTeams  updateTeams

Description

Updates the specified Team objects.

Usage

dfp_updateTeams(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- `request_data`: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a data.frame
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateTeamsResponse

See Also

Google Documentation for updateTeams

Examples

```r
## Not run:
res <- dfp_updateTeams(request_data)
## End(Not run)
```
dfp_updateTrafficAdjustments

updateTrafficAdjustments

Description

Saves all of the provided traffic adjustments. If there is already a TrafficForecastAdjustment saved for the same [@link TrafficTimeSeriesFilterCriteria], the pre-existing TrafficForecastAdjustment will be completely replaced with the submitted TrafficForecastAdjustment. This method is only available when MAKE_TRAFFIC_FORECAST_ADJUSTMENTS_IN_BULK is enabled in the global settings on your network. Saves all of the provided traffic adjustments. If there is already a TrafficForecastAdjustment saved for the same TrafficTimeSeriesFilterCriteria, the pre-existing TrafficForecastAdjustment will be completely replaced with the submitted TrafficForecastAdjustment. This method is only available when MAKE_TRAFFIC_FORECAST_ADJUSTMENTS_IN_BULK is enabled in the global settings on your network. Saves all of the provided traffic adjustments. If there is already a TrafficForecastAdjustment saved for the same TrafficTimeSeriesFilterCriteria, the pre-existing TrafficForecastAdjustment will be completely replaced with the submitted TrafficForecastAdjustment. This method is only available when MAKE_TRAFFIC_FORECAST_ADJUSTMENTS_IN_BULK is enabled in the global settings on your network.

Usage

dfp_updateTrafficAdjustments(request_data, as_df = TRUE, verbose = FALSE)

Arguments

- request_data: a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- as_df: a boolean indicating whether to attempt to parse the result into a data.frame
- verbose: a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateTrafficAdjustmentsResponse

See Also

Google Documentation for updateTrafficAdjustments

Examples

```r
## Not run:
res <- dfp_updateTrafficAdjustments(request_data)

## End(Not run)
```
**Description**

Updates the specified User objects.

**Usage**

```r
dfp_updateUsers(request_data, as_df = TRUE, verbose = FALSE)
```

**Arguments**

- `request_data`: a list or `data.frame` of data elements to be formatted for a SOAP request (XML format, but passed as character string)
- `as_df`: a boolean indicating whether to attempt to parse the result into a `data.frame`
- `verbose`: a boolean indicating whether to print the service URL and POSTed XML

**Value**

A `data.frame` or list containing all the elements of a `updateUsersResponse`

**See Also**

- [Google Documentation for updateUsers](#)

**Examples**

```r
## Not run:
res <- dfp_updateUsers(request_data)

## End(Not run)
```

---

**Description**

Updates the specified UserTeamAssociation objects.

**Usage**

```r
dfp_updateUserTeamAssociations(request_data, as_df = TRUE, verbose = FALSE)
```
Arguments

request_data a list or data.frame of data elements to be formatted for a SOAP request (XML format, but passed as character string)

as_df a boolean indicating whether to attempt to parse the result into a data.frame

verbose a boolean indicating whether to print the service URL and POSTed XML

Value

a data.frame or list containing all the elements of a updateUserTeamAssociationsResponse

See Also

Google Documentation for updateUserTeamAssociations

Examples

```r
## Not run:
res <- dfp_updateUserTeamAssociations(request_data)
## End(Not run)
```

Description

DoubleClick for Publishers API from R

Details

The DoubleClick for Publishers (DFP) API (recently renamed to Google Ad Manager) consists of roughly 50 services. Each service is written to handle a particular set of operations within the API and grouped.

The official documentation, provided by Google, is available at: https://developers.google.com/ad-manager/api/rel_notes

This package attempts to scrape the functionality and documentation from the references and WSDL to provide an interface via R. Each function has its own documentation, but additional material can be found in the README on GitHub
Index

dfp_auth, 7
dfp_createActivities, 8
dfp_createActivityGroups, 9
dfp_createAdExclusionRules, 10
dfp_createAdRules, 11
dfp_createAdUnits, 12
dfp_createAudienceSegments, 13
dfp_createBaseRates, 14
dfp_createCdnConfigurations, 15
dfp_createCompanies, 16
dfp_createContacts, 17
dfp_createContentBundles, 18
dfp_createCreatives, 19
dfp_createCreativeSet, 20
dfp_createCreativeWrappers, 21
dfp_createCustomFieldOptions, 22
dfp_createCustomFields, 23
dfp_createCustomTargetingKeys, 24
dfp_createCustomTargetingValues, 25
dfp_createDaiAuthenticationKeys, 26
dfp_createExchangeRates, 27
dfp_createLabels, 28
dfp_createLineItemCreativeAssociations, 29
dfp_createLineItems, 30
dfp_createLiveStreamEvents, 31
dfp_createMobileApplications, 32
dfp_createNativeStyles, 33
dfp_createOrders, 34
dfp_createPackages, 35
dfp_createPlacements, 36
dfp_createPremiumRates, 37
dfp_createProductPackageItems, 38
dfp_createProductPackages, 39
dfp_createProductTemplates, 40
dfp_createProposalLineItems, 41
dfp_createProposals, 42
dfp_createRateCards, 43
dfp_createTeams, 44
dfp_createUsers, 45
dfp_createUserTeamAssociations, 46
dfp_date_to_list, 47
dfp_full_report_wrapper, 47
dfp_getActivitiesByStatement, 48
dfp_getActivityGroupsByStatement, 49
dfp_getAdExclusionRulesByStatement, 50
dfp_getAdRulesByStatement, 51
dfp_getAdSpotsByStatement, 52
dfp_getAdUnitsByStatement, 53
dfp_getAdUnitSizesByStatement, 54
dfp_getAllNetworks, 55
dfp_getAllRoles, 56
dfp_getAudienceSegmentsByStatement, 57
dfp_getAvailabilityForecast, 58
dfp_getAvailabilityForecastById, 59
dfp_getBaseRatesByStatement, 60
dfp_getCdnConfigurationsByStatement, 61
dfp_getCmsMetadataKeysByStatement, 62
dfp_getCmsMetadataValuesByStatement, 63
dfp_getCompaniesByStatement, 64
dfp_getContactsByStatement, 65
dfp_getContentBundlesByStatement, 66
dfp_getContentByStatement, 67
dfp_getContentByStatementAndCustomTargetingValue, 68
dfp_getCreativesByStatement, 69
dfp_getCreativeSetsByStatement, 70
dfp_getCreativeTemplatesByStatement, 71
dfp_getCreativeWrappersByStatement, 72
dfp_getCurrentNetwork, 73
dfp_getCurrentUser, 74
dfp_getCustomFieldOption, 75
dfp_getCustomFieldsByStatement, 76
dfp_getCustomTargetingKeysByStatement, 77
dfp_getCustomTargetingValuesByStatement, 78
dfp_getDaiAuthenticationKeysByStatement, 79
dfp_getDeliveryForecast, 80
dfp_getDeliveryForecastByIds, 81
dfp_getExchangeRatesByStatement, 82
dfp_getLabelsByStatement, 83
dfp_getLineItemCreativeAssociationsByStatement, 84
dfp_getLineItemsByStatement, 85
dfp_getLineItemTemplatesByStatement, 86
dfp_getLiveStreamEventsByStatement, 87
dfp_getMarketplaceCommentsByStatement, 88
dfp_getMobileApplicationsByStatement, 89
dfp_getNativeStylesByStatement, 90
dfp_getOrdersByStatement, 91
dfp_getPackagesByStatement, 92
dfp_getPlacementsByStatement, 93
dfp_getPremiumRatesByStatement, 94
dfp_getPreviewUrl, 95
dfp_getPreviewUrlsForNativeStyles, 96
dfp_getProductPackageItemsByStatement, 96
dfp_getProductPackagesByStatement, 97
dfp_getProductsByStatement, 98
dfp_getProductTemplatesByStatement, 99
dfp_getProposalLineItemsByStatement, 100
dfp_getProposalsByStatement, 101
dfp_getRateCardsByStatement, 103
dfp_getReconciliationLineItemReportsByStatement, 104
dfp_getReconciliationOrderReportsByStatement, 105
dfp_getReconciliationReportRowsByStatement, 106
dfp_getReconciliationReportsByStatement, 107
dfp_getReportDownloadUrl, 48, 108, 150
dfp_getReportDownloadUrlWithOptions, 110
dfp_getReportJobStatus, 48, 111
dfp_getSavedQueriesByStatement, 112
dfp_getSuggestedAdUnitsByStatement, 113
dfp_getTargetingPresetsByStatement, 114
dfp_getTeamsByStatement, 115
dfp_getTrafficAdjustmentsByStatement, 116
dfp_getUsersByStatement, 117
dfp_getUserTeamAssociationsByStatement, 118
dfp_getWorkflowRequestsByStatement, 119
dfp_hasCustomPacingCurve, 120
dfp_makeTestNetwork, 121
dfp_performAdExclusionRuleAction, 122
dfp_performAdRuleAction, 123
dfp_performAdUnitAction, 124
dfp_performAudienceSegmentAction, 124
dfp_performBaseRateAction, 125
dfp_performCdnConfigurationAction, 126
dfp_performContentBundleAction, 127
dfp_performCreativeWrapperAction, 127
dfp_performCustomFieldAction, 128
dfp_performCustomTargetingKeyAction, 129
dfp_performCustomTargetingValueAction, 130
dfp_performDaiAuthenticationKeyAction, 130
dfp_performExchangeRateAction, 131
dfp_performLabelAction, 132
dfp_performLineItemAction, 133
dfp_performLineItemCreativeAssociationAction, 134
dfp_performLiveStreamEventAction, 134
dfp_performMobileApplicationAction, 135
dfp_performProposalAction, 136
dfp_performProposalLineItemAction, 137
dfp_performProductAction, 137
dfp_performPlacementAction, 138
dfp_performProductPageAction, 139
dfp_performProductPackageItemAction, 140
dfp_performProductTempalteAction, 141
dfp_performProposalAction, 142
dfp_performProposalLineItemAction, 143
dfp_performRateCardAction, 144
dfp_performReconciliationOrderReportAction, 144
dfp_performSuggestedAdUnitAction, 145
dfp_performTeamAction, 146
dfp_performUserAction, 147
dfp_performUserTeamAssociationAction, 148
dfp_performWorkflowRequestAction, 148
dfp_registerSessionsForMonitoring, 149
dfp_report_url_to_dataframe, 150
dfp_runReportJob, 48, 151
dfp_select, 152
dfp_updateActivities, 153
dfp_updateActivityGroups, 154
dfp_updateAdExclusionRules, 155
dfp_updateAdRules, 156
dfp_updateAdUnits, 156
dfp_updateAudienceSegments, 157
dfp_updateBaseRates, 158
dfp_updateCdnConfigurations, 159
dfp_updateCompanies, 159
dfp_updateContacts, 160
dfp_updateContentBundles, 161
dfp_updateCreatives, 162
dfp_updateCreativeSet, 162
dfp_updateCreativeWrappers, 163
dfp_updateCustomFieldOptions, 164
dfp_updateCustomFields, 165
dfp_updateCustomTargetingKeys, 165
dfp_updateCustomTargetingValues, 166
dfp_updateDaiAuthenticationKeys, 167
dfp_updateExchangeRates, 168
dfp_updateLabels, 168
dfp_updateLineItemCreativeAssociations, 169
dfp_updateLineItems, 170
dfp_updateLiveStreamEvents, 171
dfp_updateMobileApplications, 171
dfp_updateNativeStyles, 172
dfp_updateNetwork, 173
dfp_updateOrders, 174
dfp_updatePackages, 174
dfp_updatePlacements, 175
dfp_updatePremiumRates, 176
dfp_updateProductPackageItems, 177
dfp_updateProductPackages, 177
dfp_updateProducts, 178
dfp_updateProductTemplates, 179
dfp_updateProposalLineItems, 180
dfp_updateProposals, 180
dfp_updateRateCards, 181
dfp_updateReconciliationLineItemReports, 182
dfp_updateReconciliationOrderReports, 183
dfp_updateReconciliationReportRows, 183
dfp_updateReconciliationReports, 184
dfp_updateTeams, 185
dfp_updateTrafficAdjustments, 186
dfp_updateUsers, 187
dfp_updateUserTeamAssociations, 187
rdfp, 188
rdfp-package (rdfp), 188

Startup, 8
Token2.0, 8