Package 'ganalytics'

October 13, 2022

Title Interact with 'Google Analytics'

Version 0.10.7

Date 2019-03-02

Description Functions for querying the 'Google Analytics' core reporting, real-time, multi-channel funnel and management APIs, as well as the 'Google Tag Manager' (GTM) API. Write methods are also provided for the management and GTM APIs so that you can change tag, property or view settings, for example. Define reporting queries using natural R expressions instead of being concerned as much about API technical intricacies like query syntax, character code escaping, and API limitations.

URL https://github.com/jdeboer/ganalytics

BugReports https://github.com/jdeboer/ganalytics/issues

License MIT + file LICENCE

Depends R(>= 3.3.0)

Imports assertthat (>= 0.2.0), httpuv, httr (>= 1.0.0), googleAnalyticsR (>= 0.6.0), jsonlite, lazyeval, lubridate, methods, plyr, R6, rvest, stringr (>= 1.0), XML, xml2, scales, selectr, tibble

Suggests devtools, ggplot2, knitr, rmarkdown, testthat, dplyr, purrr, tidyr, spelling, covr

Collate 'Bool-generics.R' 'utils.R' 'Segment-generics.R' 'GaApiRequest.R' 'meta.R' 'globaldata.R' 'Expr-generics.R' 'operand-classes.R' 'comparator-classes.R' 'var-classes.R' 'expr-classes.R' 'segment-classes.R' 'Bool-methods.R' 'Comparator-generics.R' 'comparator-coerce.R' 'Comparator-methods.R' 'TableFilter-generics.R' 'table-filter-classes.R' 'var-list-classes.R' 'query-classes.R' 'Query-generics.R' 'Creds-methods.R' 'Date-generics.R' 'date-coerce.R' 'ga-api-classes.R' 'management-api-classes.R' 'Date-methods.R' 'operand-coerce.R' 'Operand-generics.R' 'Operand-methods.R' 'Var-list-generics.R' 'Var-generics.R' 'var-list-coerce.R' 'var-coerce.R' 'Var-methods.R'

'expr-coerce.R' 'Expr-methods.R' 'GaGetCoreReport.R'
'GaView-generics.R' 'view-coerce.R' 'GaView-methods.R'
'ga-api-coerce.R' 'GetGa-methods.R' 'Query-methods.R'
'segment-coerce.R' 'Sequence-generics.R' 'Segment-methods.R'
'Sequence-methods.R' 'table-filter-coerce.R'
'TableFilter-methods.R' 'Var-list-methods.R' 'deprecated.R'
'ganalytics-package.R' 'googleAnalyticsR-coerce.R'
'gtm-api-classes.R' 'sequence.R' 'zzz.R'

VignetteBuilder knitr

LazyData TRUE

RoxygenNote 6.1.1

Encoding UTF-8

Language en-GB

NeedsCompilation no

Author Johann de Boer [aut, cre]

Maintainer Johann de Boer <johanndeboer@gmail.com>

Repository CRAN

Date/Publication 2019-03-02 09:20:03 UTC

R topics documented:

And *And*

Description

Logical AND of two or more expressions.

Usage

```
And(object, ...)

## S4 method for signature '.compoundExpr'
And(object, ...)

## S4 method for signature '.compoundExpr,.compoundExpr'
e1 & e2
```

Arguments

object	An object to include within the ANDed expression.
...	Additional objects to include within the ANDed expression.
e1	first expression
e2	second expression

Value

An object of class andExpr

Methods (by class)

- .compoundExpr: Logical-AND of two or more expressions.
- e1 = .compoundExpr,e2 = .compoundExpr: Logical-AND of two expressions.

See Also

Other boolean functions: Not, Or, xor

Examples

```
purchased_on_mobile <- Expr(~deviceCategory == "mobile") & Expr(~transactions > 0)
```

Cohort *Cohort*

Description

Get or define a cohort.

Set a cohort of a query.

Usage

```
Cohort(object, value, type)

Cohort(object) <- value
```

Arguments

object	The object to get or set the cohorts of.
value	The value to set the object's cohorts to.
type	The type of cohort. Only the default of "FIRST_VISIT_DATE" is currently valid.

See Also

Other date range functions: DateRange, EndDate, SplitDateRange, StartDate

Other date range functions: DateRange, EndDate, SplitDateRange, StartDate

---

Comparator *Comparator*

Description

Get the comparator used in an expression or create a comparator object.

Set the comparator of an expression.

Usage

```
Comparator(object, ...)

Comparator(object) <- value

## S4 method for signature '.expr'
Comparator(object)

## S4 replacement method for signature '.expr'
Comparator(object) <- value
```

Arguments

object	The object to be coerced to a '.comparator' subclass or the expression object of which to obtain its comparator.
...	Used by certain methods.
value	The value to set the comparator to.

Value

An object belonging to the superclass .comparator

Methods (by class)

- .expr: Return the comparator used within the supplied conditional expression.
- .expr: Replace the comparator of the supplied conditional expression.

See Also

Other comparator functions: comparators

Other comparator functions: comparators

---

comparators *Comparison operators*

Description

Binary operators used to define Google Analytics filters and segments.

Usage

```
var %starts_with% operand

var %ends_with% operand

var %contains% operand

var %matches% operand

var %between% operand

x %in% table

## S4 method for signature '.var,.dimOperand'
var %matches% operand

## S4 method for signature '.var,.dimOperand'
var %starts_with% operand

## S4 method for signature '.var,.dimOperand'
var %ends_with% operand

## S4 method for signature '.var,.dimOperand'
var %contains% operand

## S4 method for signature '.var,.operand'
var %between% operand

## S4 method for signature '.var,.operand'
x %in% table

## S4 method for signature '.var,.operand'
e1 == e2
```

```
## S4 method for signature '.var,.operand'
e1 != e2

## S4 method for signature '.var,.metOperand'
e1 > e2

## S4 method for signature '.var,.metOperand'
e1 < e2

## S4 method for signature '.var,.metOperand'
e1 >= e2

## S4 method for signature '.var,.metOperand'
e1 <= e2
```

Arguments

var	The name of a single Google Analytics dimension or metric, as a character string or a .var object generated with Var.
operand	An operand appropriate to the selected var and comparison operator. A vector usually of length-one, or exactly length-two in the case of %between%, or at least length-two in the case of %in%. Usually either a character string or numeric value.
x	A dimension.
table	A vector of possible values within that dimension.
e1	A dimension or metric.
e2	An operand object of length-one.

Value

an expr object.

%starts_with%

A condition where the dimension (LHS) matches values that start with the character string given by the operand (RHS).

%ends_with%

A condition where the dimension (LHS) matches values that end with the character string given by the operand (RHS).

%contains%

A condition where the dimension (LHS) matches values that contain the character string given by the operand (RHS).

%matches%

A condition where the dimension (LHS) matches a regular expression given by the operand (RHS).

%between%

A condition where the var (LHS) is within the lower and upper bounds specified by first and second vector value (respectively) of the operand (RHS).

%in%

A condition where the dimension (LHS) matches one of the values in the vector specified by the operand (RHS).

Equal-to (==)

Do the values on the left and right match exactly.

Not equal-to (!=)

Do the values on the left and right not match.

Greater-than (>)

Is the value on the left greater than the value on the right.

Less-than (<)

Is the value on the left less than the value on the right.

Greater-than-or-equal-to (>=)

Is the value on the left greater than or equal to the value on the right.

Less-than-or-equal-to (<=)

Is the value on the left less than or equal to the value on the right.

See Also

Other comparator functions: Comparator

Examples

```
Expr(~PagePath %starts_with% "/products")
Expr(~PagePath %ends_with% "/index.html")
Expr(~PagePath %contains% "thank-you")
Expr(~PagePath %matches% "*.thank[\\-_]?you.*")
Expr(~transactionRevenue %between% c(200, 500))
Expr(~browser %in% c("Chrome", "Firefox"))
Expr(~productName == "apple")
Expr(~bounces == 0)
```

```
Expr(~deviceCategory != "tablet")
Expr(~sessionDuration != 0)
Expr(~pageviews > 100)
Expr(~exits < 100)
```

DateRange *DateRange*

Description

Get the date range.

Set the date range.

Usage

```
DateRange(object, endDate)

DateRange(object) <- value

## S4 method for signature 'character'
StartDate(object)

## S4 method for signature 'character'
EndDate(object)

## S4 method for signature 'dateRange'
StartDate(object)

## S4 method for signature 'dateRange'
EndDate(object)

## S4 method for signature 'Interval'
StartDate(object)

## S4 method for signature 'Interval'
EndDate(object)

## S4 method for signature '.standardQuery'
StartDate(object)

## S4 method for signature '.standardQuery'
EndDate(object)

## S4 method for signature 'gaView'
StartDate(object)

## S4 method for signature 'gaView'
```

```
EndDate(object)

## S4 replacement method for signature 'dateRange'
StartDate(object) <- value

## S4 replacement method for signature 'dateRange'
EndDate(object) <- value

## S4 replacement method for signature '.standardQuery'
StartDate(object) <- value

## S4 replacement method for signature '.standardQuery'
EndDate(object) <- value

## S4 method for signature 'ANY,ANY'
DateRange(object, endDate)

## S4 method for signature 'ANY,missing'
DateRange(object)

## S4 method for signature '.standardQuery,missing'
DateRange(object)

## S4 replacement method for signature '.standardQuery'
DateRange(object) <- value

## S4 method for signature 'gaView,ANY'
DateRange(object)

## S4 replacement method for signature 'ANY'
DateRange(object) <- value
```

Arguments

object	The start date of the date range or a object to coerce to a date range. Alternatively, a .query object to replace the date range of.
endDate	The end date of the date range. Alternatively, if object is a .query object, then endDate is the replacement date range.
value	The replacement date range.

Methods (by class)

- character: Coerce a character vector into a Google Analytics date object.
- character: Coerce a character vector into a Google Analytics date object.
- dateRange: Return the start dates of a date range vector.
- dateRange: Return the end dates of a date range vector.
- Interval: Return the start dates of a date range vector.

- `Interval`: Return the end dates of a date range vector.
- `.standardQuery`: Return the start dates of a query's date range vector.
- `.standardQuery`: Return the end dates of a query's date range vector.
- `gaView`: Get the date when a view first started receiving hits.
- `gaView`: Get the last day a view received hits.
- `dateRange`: Set a new start date for a date range.
- `dateRange`: Set a new end date for a date range.
- `.standardQuery`: Set a new start date for a query.
- `.standardQuery`: Set a new end date for a query.
- `object = ANY,endDate = ANY`: generates a date range object using the supplied vectors of start date and end dates.
- `object = ANY,endDate = missing`: Returns the date range of the given query or coerces the supplied object into a dateRange.
- `object = .standardQuery,endDate = missing`: Returns the date range of the given query.
- `.standardQuery`: Modify the date range of the given query.
- `object = gaView,endDate = ANY`: Returns the maximum date range of when a view has been receiving hits.
- `ANY`: Change the date range of the date range object using the dates supplied in a vector of length 2, where the first element is the start date and second being the end date.

See Also

Other date range functions: `Cohort`, `EndDate`, `SplitDateRange`, `StartDate`

Other date range functions: `Cohort`, `EndDate`, `SplitDateRange`, `StartDate`

`Dimensions` *Dimensions*

Description

Get the dimensions of the object.

Set the dimensions for the object.

Usage

```
Dimensions(object, ...)

Dimensions(object) <- value

## S4 method for signature 'ANY'
Dimensions(object, ...)
```

```
## S4 method for signature '.query'
Dimensions(object)

## S4 replacement method for signature '.query'
Dimensions(object) <- value
```

Arguments

object	An object to be coerced to a list of dimensions.
...	Other dimensions to add to the returned list, or if object is a query object, the replacement dimensions.
value	The replacement dimensions for the supplied object.

Methods (by class)

- ANY: Coerces the supplied character vector or list into a vector of Google Analytics dimensions.
- .query: Returns the dimensions used within the supplied query.
- .query: Replace the dimensions of the query.

See Also

Other query object functions: MaxResults, Metrics, SamplingLevel, SortBy, TableFilter

Other query object functions: MaxResults, Metrics, SamplingLevel, SortBy, TableFilter

---

DynSegment *DynSegment*

Description

Combine one or more segment condition filters and/or sequence filters into a gaDynSegment that is scoped to either 'user' or 'session' level.

Usage

```
DynSegment(object, ..., name = character(0))

## S4 method for signature 'ANY'
DynSegment(object, ..., name = character(0))

## S4 method for signature 'gaDynSegment'
DynSegment(object)
```

Arguments

object	The first filter to include in the segment definition.
...	Additional filters to include in the segment definition, if needed.
name	An optional name given to the dynamic segment.

Details

Segment filter are either sequential or non-sequential conditions. Sequential and non-sequential conditions can be combined using this function.

Value

A gaDynSegment object.

Methods (by class)

- ANY: Defines a list of filters from one or more expressions applied using the specified scope.
- gaDynSegment: Returns itself.

See Also

Other dynamic segment functions: Exclude, Include, IsNegated, PerHit, PerProduct, PerSession, PerUser, ScopeLevel, SegmentConditionFilter, Segments, Segment

Examples

```
return_shoppers <- SegmentConditionFilter(
  Expr(~transactions > 1, metricScope = "perUser"),
  scope = "users"
)
watched_video_then_purchased <- Sequence(
  Expr(~eventCategory == "video") & Expr(~eventAction == "play"),
  Later(Expr(~transactions > 0))
)
```

EndDate *EndDate*

Description

Get the end date of the date range.

Set the endDate of the date range.

Usage

```
EndDate(object, value)

EndDate(object) <- value
```

Arguments

object	Object to get end date of.
value	Value to set end date of object to.

See Also

Other date range functions: Cohort, DateRange, SplitDateRange, StartDate

Other date range functions: Cohort, DateRange, SplitDateRange, StartDate

---

Exclude *Exclude*

Description

Set the negation flag of a segment filter to TRUE.

Usage

```
Exclude(object, ..., scope)

## S4 method for signature 'ANY'
Exclude(object, ..., scope)
```

Arguments

object	A segment condition or sequence filter to include.
...	Additional segment conditions to include.
scope	Optional scope, "users" or "sessions" (default).

Value

A .gaSegmentFilter object with its negate slot set to TRUE.

Methods (by class)

- ANY: Define an exclude segment filter using the supplied expressions.

See Also

Other dynamic segment functions: DynSegment, Include, IsNegated, PerHit, PerProduct, PerSession, PerUser, ScopeLevel, SegmentConditionFilter, Segments, Segment

Examples

```
exclude_one_time_shoppers <- Exclude(
  Expr(~transactions == 1, metricScope = "perUser"),
  scope = "users"
)
```

Expr *Expr*

Description

Define a Google Analytics expression.

Usage

```
Expr(object, comparator, operand, metricScope = "")

## S4 method for signature '.expr,ANY'
Expr(object)

## S4 method for signature 'formula,ANY'
Expr(object, metricScope)

## S4 method for signature 'character,character'
Expr(object, comparator, operand,
  metricScope = "")
```

Arguments

object	A dimension or metric variable, or another object to be coerced to an .expr object.
comparator	The comparator to use for the expression.
operand	The operand to use for the expression.
metricScope	Optional scope to use in the case of metric variables for segmentation. Possible values include "perUser" or "perSession".

Methods (by class)

- object = .expr,comparator = ANY: Returns itself.
- object = formula,comparator = ANY: Use non-standard formula evaluation to define an expression. Accepts a formula in the form of: ~ <variable> <comparator> <operand> where only the <operand> is evaluated.
- object = character,comparator = character: Return an expression composed of the supplied variable, comparator and operand arguments.

See Also

Other expression generators: GaExpr, McfExpr, RtExpr

Examples

```
source_google <- Expr(~source == "google")
source_google <- Expr("source", "==", "google")
bounces <- Expr("bounces", ">", 0)
```

First *First*

Description

If used at the beginning of a sequence, indicates that this step must match the first interaction of included sessions and users within the select date range. First expressly means 'first interaction' within the date range.

Usage

```
First(object, ...)
```

Arguments

object	An expression that should be at the start of a sequence expression.
...	Any other expressions that should immediately follow the first expression.

Value

A gaSegmentSequenceStep object, with the immediate flag set.

See Also

Sequence

Other sequence segment functions: Later, Sequence, Then

Examples

```
session_starts_from_home <- First(Expr(~pagepath == "/"))
```

GaAccount *GA Account*

Description

Get a GA account.

Usage

```
GaAccount(id = NULL, creds = get_creds())
```

Arguments

id	ID of the GA account to get
creds	The Google APIs credentials to use.

GaAccounts *GTM Accounts*

Description

Get the collection of GA accounts accessible to the user with the credentials supplied by creds..

Usage

```
GaAccounts(creds = get_creds())
```

Arguments

creds	The Google APIs credentials to use.

GaAccountSummaries *GTM Account Summaries*

Description

Get a collection of Google Analytics account summaries.

Usage

```
GaAccountSummaries(creds = get_creds())
```

Arguments

creds	The Google APIs credentials to use.

GaAccountSummary *GA Account Summary*

Description

Get a Google Analytics account summary resource.

Usage

```
GaAccountSummary(id = NULL, creds = get_creds())
```

Arguments

id	ID of the GA account to get a summary of.
creds	The Google APIs credentials to use.

GaCreds *Authentication credentials for Google Analytics API queries.*

Description

Get or set the authentication credentials for a Google Analytics query object.

Usage

```
GaCreds(object = "GANALYTICS", value = NULL, ...)

GaCreds(object) <- value

## S4 method for signature '.query,list'
GaCreds(object, value)

## S4 method for signature '.query,ANY'
GaCreds(object)

## S4 method for signature 'character,ANY'
GaCreds(object = "GANALYTICS", value = NULL,
  ...)

## S4 method for signature 'missing,ANY'
GaCreds(object = "GANALYTICS", value = NULL,
  ...)

## S4 replacement method for signature '.query,list'
GaCreds(object) <- value
```

Arguments

object	The object to get the credentials from.
value	The replacement credentials for the supplied query object.
...	other arguments pass to GoogleApiCreds.

Methods (by class)

- object = .query,value = list: Return the query with the supplied authentication credentials supplied.
- object = .query,value = ANY: Return the credentials used within the supplied query.
- object = character,value = ANY: Create a set of authentication credentials using the supplied application name.
- object = missing,value = ANY: Return default authentication credentials.
- object = .query,value = list: Replace the authentication credentials of a query.

GaExpr *GaExpr*

Description

Create a Core Reporting API expression.

Usage

```
GaExpr(object, comparator, operand, metricScope = "")

## S4 method for signature 'character,character'
GaExpr(object, comparator, operand,
  metricScope = "")
```

Arguments

object	A dimension or metric variable, or another object to be coerced to an .expr object.
comparator	The comparator to use for the expression.
operand	The operand to use for the expression.
metricScope	Optional scope to use in the case of metric variables for segmentation. Possible values include "perUser" or "perSession".

Methods (by class)

- object = character,comparator = character: Return a Google Analytics expression using the supplied variable, operator and operand. bounces <- GaExpr("bounces", ">", 0)

See Also

Other expression generators: Expr, McfExpr, RtExpr

Examples

```
myQuery <- GaQuery(view = 123456789)
source_matches_google <- GaExpr("source", "~", "google")
TableFilter(myQuery) <- source_matches_google
```

GaMetaUpdate *GaMetaUpdate Update the metadata file Update the package system file containing metadata about valid dimensions and metrics, etc. This function should be used prior to package build.*

Description

GaMetaUpdate Update the metadata file Update the package system file containing metadata about valid dimensions and metrics, etc. This function should be used prior to package build.

Usage

```
GaMetaUpdate(creds = get_creds())
```

Arguments

creds	Google Analytics OAuth 2.0 credentials object.

Value

a data.frame

GaQuery *GaQuery.*

Description

Create a ganalytics query object

Usage

```
GaQuery(view = NA, creds = get_creds(), startDate = Sys.Date() - 8,
  endDate = Sys.Date() - 2, metrics = "ga:sessions",
  dimensions = "ga:date", sortBy = NULL, filters = NULL,
  segments = NULL, cohorts = NULL, samplingLevel = "DEFAULT",
  maxResults = kGaMaxResults)
```

Arguments

view	view id to use
creds	authentication credentials object created using GoogleApiCreds()
startDate	start date
endDate	end date
metrics	character vector of metrics
dimensions	character vector of dimensions
sortBy	a sort by object
filters	a filters object
segments	a segment object or list of segments
cohorts	a cohort object or a list of cohorts
samplingLevel	either "DEFAULT", "HIGHER_PRECISION" or "FASTER"
maxResults	the maximum number of results to return, up to 1,000,000

GaUserSegment *User Segments*

Description

Get a user or system defined segment.

Usage

```
GaUserSegment(id = NULL, definition = NA, creds = get_creds())
```

Arguments

id	ID of the user or system segment to get
definition	The definition to use if an ID is not provided.
creds	The Google APIs credentials to use.

GaUserSegments *Get a collection of user and system defined segments.*

Description

Returns a collection object of user and system defined segments available under the supplied user credentials.

Usage

```
GaUserSegments(creds = get_creds())
```

Arguments

creds	The Google APIs credentials to use.

ga_view_selector *ga_view_selector*

Description

A menu user-interface for selecting a Google Analytics view.

Get the view ID of the query.

Set the view ID for the query.

Usage

```
ga_view_selector(creds = GoogleApiCreds(), with_gui = FALSE)

GaView(object, value)

GaView(object) <- value

## S4 method for signature 'gaProperty,missing'
GaView(object)

## S4 method for signature 'gaAccount,missing'
GaView(object)

## S4 method for signature 'ANY,missing'
GaView(object)

## S4 method for signature '.query,missing'
GaView(object)
```

```
## S4 method for signature '.query,ANY'
GaView(object, value)

## S4 replacement method for signature '.query'
GaView(object) <- value
```

Arguments

creds	Optional. An OAuth2.0 credentials object to use for the request.
with_gui	Optional. Boolean value indicating whether to use a GUI for the menu. Default is FALSE.
object	An object to coerce to a gaView object or to get the gaView of, such as a query, default view of a web property, or the default view of the first web property in a Google Analytics account.
value	The optional replacement view if the object supplied is a query, in which case GaView will return the modified query.

Value

A gaView object.

Methods (by class)

- object = gaProperty,value = missing: Select the default view of the property.
- object = gaAccount,value = missing: Select the default view of the first listed property of the account.
- object = ANY,value = missing: Returns the ID of the supplied view, or the default view within the supplied property or the default view within the first property of the supplied account, or coerces a numeric or character into a viewId.
- object = .query,value = missing: gets the view ID of the supplied query.
- object = .query,value = ANY: Set the view of a query, returning the query with the updated view applied.
- .query: Replaces the view being used by a query.

Examples

```
## Not run:
   my_ga_account <- GaAccounts()[['60253332']]
   my_website_property <- my_ga_account$properties[['UA-60253332-2']]
   my_default_view <- GaView(my_website_property)

## End(Not run)
## Not run:
   my_ga_account <- GaAccounts()[['60253332']]
   my_default_view <- GaView(my_ga_account)

## End(Not run)
```

GetGaData *GetGaData*

Description

Fetch the data for the Google Analytics API query.

Usage

```
GetGaData(query, creds = NULL, ...)
```

Arguments

query	The query execute and returned the processed response for.
creds	The OAuth2.0 credentials to use for the request.
...	Other arguments to pass on to lower-level API functions.

GetGaData,.query-method

GetGaData Execute a ganalytics query.

Description

GetGaData Execute a ganalytics query.

Usage

```
## S4 method for signature '.query'
GetGaData(query, creds = NULL, .progress = "time",
  addViewId = FALSE, addSegmentId = FALSE)
```

Arguments

query	the query to execute.
creds	the Google APIs Project OAuth 2.0 credentials to use.
.progress	progress bar to display. use .progress = "none" to turn off.
addViewId	logical value indicating whether to add a viewID column for when more than one view has been provided.
addSegmentId	logical value indicating whether to add the name of the segment for when more than one segment has been queried.

Value

a tibble

GoogleApiCreds *Google APIs OAuth 2.0 Credentials.*

Description

Create a Google APIs OAuth2.0 credentials object

Usage

```
GoogleApiCreds(userName = Sys.getenv(paste0(appname, "_USER")),
  appCreds = NULL, cache = character(0), use_oob = FALSE,
  appname = "GOOGLE_APIS")
```

Arguments

userName	Google username email address hint
appCreds	Filename or named vector for client_id and client_secret.
cache	httr OAuth2.0 cache
use_oob	as per httr
appname	prefix of environment variables that hold the client ID and client secret.

GtmAccount *GTM Account*

Description

Get a GTM account.

Usage

```
GtmAccount(id = NULL, creds = get_creds())
```

Arguments

id	ID of the GTM account to get
creds	The Google APIs credentials to use.

GtmAccounts *GTM Accounts*

Description

Get a collection of GTM accounts.

Usage

```
GtmAccounts(creds = get_creds())
```

Arguments

creds	The Google APIs credentials to use.

Include *Include*

Description

Set the negation flag of a segment filter to FALSE.

Usage

```
Include(object, ..., scope)

## S4 method for signature 'ANY'
Include(object, ..., scope)
```

Arguments

object	A segment condition or sequence filter to include.
...	Additional segment conditions to include.
scope	Optional scope, "users" or "sessions" (default).

Value

A .gaSegmentFilter object with its negate slot set to FALSE.

Methods (by class)

- ANY: Define an include segment filter using the supplied expression.

See Also

Other dynamic segment functions: DynSegment, Exclude, IsNegated, PerHit, PerProduct, PerSession, PerUser, ScopeLevel, SegmentConditionFilter, Segments, Segment

Examples

```
return_shoppers <- Include(
  Expr(~transactions > 1, metricScope = "perUser"),
  scope = "users"
)
```

IsNegated *IsNegated*

Description

Tests whether a segment filter is negated.

Sets the negation flag of a segment filter.

Usage

```
IsNegated(object)

IsNegated(object) <- value

## S4 method for signature '.gaSegmentFilter'
IsNegated(object)

## S4 replacement method for signature '.gaSegmentFilter,logical'
IsNegated(object) <- value
```

Arguments

object	An object belonging to the superclass .gaSegmentFilter.
value	The value of the negation slot, either TRUE or FALSE.

Methods (by class)

- .gaSegmentFilter: Tests whether a segment filter is negated, i.e. used to define an exclude filter for the segment.
- object = .gaSegmentFilter,value = logical: Sets whether a segment filter should be negated, i.e. used as an exclude filter in a segment definition.

See Also

Other dynamic segment functions: DynSegment, Exclude, Include, PerHit, PerProduct, PerSession, PerUser, ScopeLevel, SegmentConditionFilter, Segments, Segment

Other dynamic segment functions: DynSegment, Exclude, Include, PerHit, PerProduct, PerSession, PerUser, ScopeLevel, SegmentConditionFilter, Segments, Segment

Examples

```
exclude_one_time_shoppers <- Exclude(
  Expr(~transactions == 1, metricScope = "perUser"),
  scope = "users"
)
IsNegated(exclude_one_time_shoppers) # TRUE
```

IsRegEx *IsRegEx*

Description

Checks for a regular expression.

Usage

```
IsRegEx(object)

## S4 method for signature '.dimComparator'
IsRegEx(object)

## S4 method for signature '.expr'
IsRegEx(object)
```

Arguments

object	An object to check if whether a regular expression.

Value

TRUE or FALSE

Methods (by class)

- .dimComparator: Test whether the supplied comparator is for a regular expression.
- .expr: Test whether a conditional expression is using regular expression match.

Later *Later*

Description

Treat a step within a sequence as happening at any point after any preceding steps in the sequence, i.e. 'later'. 'Later' means 'followed by', but not necessarily immediately.

Usage

```
Later(object, ...)
```

Arguments

object	The expression that should precede others in the sequence.
...	Any other expressions that should follow the first one but before any others in the sequence.

Value

A gaSegmentSequenceStep object, with the immediate flag not set.

See Also

Other sequence segment functions: First, Sequence, Then

Examples

```
purchased_sometime_later <- Later(Expr(~transactions > 0))
```

MaxResults *MaxResults*

Description

Get the value set for MaxResults.

Set the maximum rows returned by a ganalytics query.

Usage

```
MaxResults(object, value)

MaxResults(object) <- value

## S4 method for signature '.query,missing'
MaxResults(object)

## S4 method for signature '.query,ANY'
MaxResults(object, value)

## S4 replacement method for signature '.query'
MaxResults(object) <- value
```

Arguments

object	A query object.
value	Replacement value for the max-results parameter of the query.

Methods (by class)

- object = .query,value = missing: Return the maximum number of rows a query is allowed to return.
- object = .query,value = ANY: Set the maximum number of rows a query is allowed to return.
- .query: Set the maximum number of rows a query is allowed to return.

See Also

Other query object functions: Dimensions, Metrics, SamplingLevel, SortBy, TableFilter

Other query object functions: Dimensions, Metrics, SamplingLevel, SortBy, TableFilter

---

McfExpr *McfExpr*

Description

Create a Multi-Channel Funnel Reporting API expression.

Usage

```
McfExpr(object, comparator, operand)

## S4 method for signature 'character,character'
McfExpr(object, comparator, operand)
```

Arguments

object	A dimension or metric variable, or another object to be coerced to an .expr object.
comparator	The comparator to use for the expression.
operand	The operand to use for the expression.

Methods (by class)

- object = character,comparator = character: Return a Multi-channel Funnel condition composed of the supplied arguments describing the variable, comparator and operator.

See Also

Other expression generators: Expr, GaExpr, RtExpr

Examples

```
myQuery <- McfQuery(view = 123456789)
source_matches_google <- McfExpr("mcf:source", "~", "google")
TableFilter(myQuery) <- source_matches_google
```

McfQuery *McfQuery.*

Description

Create a Multi-Channel Funnel Reporting API query object

Usage

```
McfQuery(view = NA, creds = get_creds(), startDate = Sys.Date() - 8,
  endDate = Sys.Date() - 2, metrics = "mcf:totalConversions",
  dimensions = "mcf:nthDay", sortBy = NULL, filters = NULL,
  samplingLevel = "DEFAULT", maxResults = kGaMaxResults)
```

Arguments

view	view id to use
creds	authentication credentials object created using GoogleApiCreds()
startDate	start date
endDate	end date
metrics	character vector of metrics
dimensions	character vector of dimensions
sortBy	a sort by object

filters	a filters object
samplingLevel	either "DEFAULT", "HIGHER_PRECISION" or "FASTER"
maxResults	the maximum number of results to return, up to 1,000,000

Metrics *Metrics*

Description

Get the metrics of the object.

Set the metrics of the object.

Usage

```
Metrics(object, ...)

Metrics(object) <- value

## S4 method for signature 'ANY'
Metrics(object, ...)

## S4 method for signature '.query'
Metrics(object)

## S4 replacement method for signature '.query'
Metrics(object) <- value
```

Arguments

object	An object to coerce to a list of metrics, or a query object to replace the metrics of.
...	Further metrics to add to the resulting list or the replacement value for the metrics of the query object (if supplied).
value	The replacement dimensions for the supplied object.

Methods (by class)

- ANY: Coerce one or more supplied objects to .metrics.
- .query: Get the list of metrics for a '.query'.
- .query: Set the metrics for a '.query' object.

See Also

Other query object functions: Dimensions, MaxResults, SamplingLevel, SortBy, TableFilter

Other query object functions: Dimensions, MaxResults, SamplingLevel, SortBy, TableFilter

Not *Not*

Description

Not inverts an expression, i.e. logical NOT.

Usage

```
Not(object)

## S4 method for signature '.comparator'
Not(object)

## S4 method for signature '.comparator'
!x

## S4 method for signature '.expr'
Not(object)

## S4 method for signature '.expr'
!x

## S4 method for signature 'orExpr'
Not(object)

## S4 method for signature 'orExpr'
!x

## S4 method for signature '.gaSegmentFilter'
Not(object)

## S4 method for signature '.gaSegmentFilter'
!x
```

Arguments

object	An object to get the logical inverse of.
x	the object to return the logical inverse of.

Methods (by class)

- .comparator: Return the inverse of the supplied comparison operator.
- .comparator: Return the inverse of the supplied comparator.
- .expr: Invert the comparator of a condition expression.
- .expr: Invert the comparator of the condition expression.

- orExpr: Invert an OR expression using De Morgan's Theorem.
- orExpr: Invert an OR expression using De Morgan's Theorem.
- .gaSegmentFilter: Invert the negation of a segment filter condition, i.e. include <-> exclude
- .gaSegmentFilter: Invert the negation of a segment filter condition, i.e. include <-> exclude

See Also

Other boolean functions: And, Or, xor

Examples

```
source_matches_google <- Expr(~source %matches% "google")
source_not_matching_google <- Not(source_matches_google)
identical(source_not_matching_google, !source_matches_google)
```

Operand *Operand*

Description

Get the operand of an expression.

Set the operand of an expression.

Usage

```
Operand(object, value)

Operand(object) <- value

## S4 method for signature '.expr'
Operand(object)

## S4 replacement method for signature '.expr'
Operand(object) <- value
```

Arguments

object	The object for which to set the operand of.
value	Character or numeric. The value to set the operand to.

Methods (by class)

- .expr: Return the operand used within the condition, or coerce the supplied value into an operand.
- .expr: Replace the operand of a condition.

Or *Or*

Description

Logical OR of two or more expressions.

Usage

```
Or(object, ...)

## S4 method for signature '.compoundExpr'
Or(object, ...)

## S4 method for signature '.compoundExpr,.compoundExpr'
e1 | e2
```

Arguments

object	An object to include within the ORed expression.
...	Additional objects to include within the ORed expression.
e1	first expression
e2	second expression

Value

An object of class orExpr.

Methods (by class)

- .compoundExpr: Logical-OR of two or more expressions.
- e1 = .compoundExpr,e2 = .compoundExpr: Logical-OR of two expressions.

Note

Google Analytics does not support ORing of ANDed expressions – Only ANDing of ORed expresisons are supported. Consider De Morgan's laws for possible ways to work around this limitation.

See Also

Other boolean functions: And, Not, xor

Examples

```
mobile_or_tablet <- Expr(~deviceCategory == "mobile") | Expr(~deviceCategory == "tablet")
converted <- Expr(~goalCompletionsAll > 0) | Expr(~transactions > 0)
```

PerHit *PerHit*

Description

Set the scope of a gaMetExpr object to hit-level, or transforms a condition filter to a sequence filter of length one (i.e. a combination of conditions for matching a single hit).

Usage

```
PerHit(object, ..., negation)

## S4 method for signature '.compoundExpr'
PerHit(object, ..., negation)

## S4 method for signature 'gaMetExpr'
PerHit(object, ..., negation)

## S4 method for signature 'formula'
PerHit(object, ..., negation)
```

Arguments

object	A gaMetExpr object to coerce to hit-level or if multiple expressions are provided, then the first expression to combine into a single step sequence filter.
...	Further expressions to be included in the filter definition if defining a sequence filter of length one.
negation	Boolean value indicating whether to negate the condition.

Value

A gaMetExpr or gaSegmentSequenceFilter.

Methods (by class)

- .compoundExpr: Create a single step sequence filter from the supplied expression.
- gaMetExpr: Set the scope of the supplied metric condition to hit-level.
- formula: Set the scope of the supplied non-standard-evaluation metric condition to hit-level.

See Also

Other dynamic segment functions: DynSegment, Exclude, Include, IsNegated, PerProduct, PerSession, PerUser, ScopeLevel, SegmentConditionFilter, Segments, Segment

Examples

```
spent_more_than_100_in_a_transaction <- PerHit(Expr(~transactionRevenue > 100))
played_intro_video <- PerHit(
  Expr(~eventCategory == "Video") &
  Expr(~eventAction == "Play") &
  Expr(~eventLabel == "Intro")
)
```

PerProduct *PerProduct*

Description

Set the scope of a gaMetExpr object to product-level.

Usage

```
PerProduct(object, negation)

## S4 method for signature 'gaMetExpr'
PerProduct(object)

## S4 method for signature 'formula'
PerProduct(object)
```

Arguments

object	A gaMetExpr object to coerce to hit-level
negation	Boolean value indicating whether to negate the condition.

Value

A gaMetExpr object.

Methods (by class)

- gaMetExpr: Set the scope of the supplied metric condition to product-level.
- formula: Set the scope of the supplied non-standard-evaluation metric condition to product-level.

See Also

Other dynamic segment functions: DynSegment, Exclude, Include, IsNegated, PerHit, PerSession, PerUser, ScopeLevel, SegmentConditionFilter, Segments, Segment

Examples

```
with_products_added_more_than_once <- PerProduct(Expr(~productAddsToCart > 1))
```

PerSession *PerSession*

Description

Set the scope of a .gaSegmentFilter or gaMetExpr object to session-level.

Usage

```
PerSession(object, ..., negation)

## S4 method for signature 'ANY'
PerSession(object, ..., negation)

## S4 method for signature 'gaSegmentSequenceStep'
PerSession(object, ..., negation)

## S4 method for signature 'gaSegmentSequenceFilter'
PerSession(object, ..., negation)

## S4 method for signature 'gaMetExpr'
PerSession(object, ..., negation)

## S4 method for signature 'formula'
PerSession(object, ..., negation)
```

Arguments

object	A .gaSegmentFilter or gaMetExpr object to coerce to session-level. Alternatively, an dimension expression or segment filter to coerce into a session scoped gaDynSegment.
...	Other filters to include in the gaDynSegment.
negation	Boolean value indicating whether to negate the condition.

Value

A gaMetExpr, .gaSegmentFilter or gaDynSegment.

Methods (by class)

- ANY: Create a session level segment filter list from the supplied expressions, interpreted as condition filters.
- gaSegmentSequenceStep: Create a session-level segment sequence filter from the supplied sequence expression.
- gaSegmentSequenceFilter: Create a session-level segment sequence filter from the supplied sequence expression.
- gaMetExpr: Set the scope of the supplied metric condition to session-level.
- formula: Set the scope of the supplied non-standard-evaluation metric condition to session-level.

Note

To define a gaDynSegment comprised of a single metric expression, wrap the metric expression in an Include or Exclude call.

See Also

Other dynamic segment functions: DynSegment, Exclude, Include, IsNegated, PerHit, PerProduct, PerUser, ScopeLevel, SegmentConditionFilter, Segments, Segment

Examples

```
spent_more_than_100_in_a_session <- PerSession(Expr(~transactionRevenue > 100))
```

PerUser *PerUser*

Description

Set the scope of a .gaSegmentFilter or gaMetExpr object to user-level.

Usage

```
PerUser(object, ..., negation)

## S4 method for signature 'ANY'
PerUser(object, ..., negation)

## S4 method for signature 'gaSegmentSequenceStep'
PerUser(object, ..., negation)

## S4 method for signature 'gaSegmentSequenceFilter'
PerUser(object, ..., negation)
```

```
## S4 method for signature 'gaMetExpr'
PerUser(object, ..., negation)

## S4 method for signature 'formula'
PerUser(object, ..., negation)
```

Arguments

object	a .gaSegmentFilter or gaMetExpr object to coerce to user-level. Alternatively, an dimension expression or segment filter to coerce into a user scoped gaDynSegment.
...	Other filters to include in the gaDynSegment.
negation	Boolean value indicating whether to negate the condition.

Value

A gaMetExpr, .gaSegmentFilter or gaDynSegment.

Methods (by class)

- ANY: Create a user-level segment filter list from the supplied expressions, each interpreted as an include segment filter.
- gaSegmentSequenceStep: Create a user-level segment sequence filter from the supplied sequence expression.
- gaSegmentSequenceFilter: Create a user-level segment sequence filter from the supplied sequence expression.
- gaMetExpr: Set the scope of the supplied metric condition to user-level.
- formula: Set the scope of the supplied non-standard-evaluation metric condition to user-level.

Note

To define a gaDynSegment comprised of a single metric expression, wrap the metric expression in an Include or Exclude call.

See Also

Other dynamic segment functions: DynSegment, Exclude, Include, IsNegated, PerHit, PerProduct, PerSession, ScopeLevel, SegmentConditionFilter, Segments, Segment

Examples

```
spent_more_than_100_per_user <- PerUser(Expr(~transactionRevenue > 100))
```

RtExpr *RtExpr*

Description

Create a Real-Time Reporting API expression.

Usage

```
RtExpr(object, comparator, operand)

## S4 method for signature 'character,character'
RtExpr(object, comparator, operand)
```

Arguments

object	A dimension or metric variable, or another object to be coerced to an .expr object.
comparator	The comparator to use for the expression.
operand	The operand to use for the expression.

Methods (by class)

- object = character,comparator = character: Define a Real-Time Reporting condition using the arguments describing the variable, comparator and operand.

See Also

Other expression generators: Expr, GaExpr, McfExpr

Examples

```
myQuery <- RtQuery(view = 123456789)
source_matches_google <- RtExpr("rt:source", "~", "google")
TableFilter(myQuery) <- source_matches_google
```

RtQuery *RtQuery.*

Description

Create a Real-Time reporting API query object

Usage

```
RtQuery(view = NA, creds = get_creds(), metrics = "rt:pageviews",
  dimensions = "rt:minutesAgo", sortBy = NULL, filters = NULL,
  maxResults = kGaMaxResults)
```

Arguments

view	view id to use
creds	authentication credentials object created using GoogleApiCreds()
metrics	character vector of metrics
dimensions	character vector of dimensions
sortBy	a .sortBy object
filters	a .tableFilter object
maxResults	the maximum number of results to return, up to 1,000,000

SamplingLevel *SamplingLevel*

Description

Get the sampling level.

Set the sampling level for a ganalytics query.

Usage

```
SamplingLevel(object, value)

SamplingLevel(object) <- value

## S4 method for signature '.standardQuery,missing'
SamplingLevel(object)

## S4 method for signature '.standardQuery,ANY'
SamplingLevel(object, value)
```

```
## S4 replacement method for signature '.standardQuery'
SamplingLevel(object) <- value

## S4 method for signature 'data.frame,ANY'
SamplingLevel(object)
```

Arguments

object	The query or response to check the sampling level of.
value	Optional. If object is a query, then value can be provided to set the sampling level to of that query, in which case an updated query object will be returned.

Methods (by class)

- object = .standardQuery,value = missing: Return what level the sampling level of the query has been set to.
- object = .standardQuery,value = ANY: Set the sampling level of the query.
- .standardQuery: Set the sampling level of the query.
- object = data.frame,value = ANY: Return details about any sampling that was applied in the response of the query.

See Also

Other query object functions: Dimensions, MaxResults, Metrics, SortBy, TableFilter

Other query object functions: Dimensions, MaxResults, Metrics, SortBy, TableFilter

---

ScopeLevel *ScopeLevel*

Description

Get or set the scope level of a .gaSegmentFilter or gaMetExpr.

Set the scope level of a .gaSegmentFilter or a gaMetExpr.

Usage

```
ScopeLevel(object, value)

ScopeLevel(object) <- value

## S4 method for signature 'gaSegMetExpr,missing'
ScopeLevel(object)

## S4 method for signature 'gaSegMetExpr,character'
ScopeLevel(object, value)
```

```
## S4 replacement method for signature 'gaMetExpr,character'
ScopeLevel(object) <- value

## S4 method for signature '.gaSegmentFilter,missing'
ScopeLevel(object)

## S4 method for signature '.gaSegmentFilter,character'
ScopeLevel(object, value)

## S4 replacement method for signature '.gaSegmentFilter,character'
ScopeLevel(object) <- value

## S4 replacement method for signature 'gaDynSegment,character'
ScopeLevel(object) <- value
```

Arguments

object	A .gaSegmentFilter or gaMetExpr object.
value	Optional new scope level to return an updated copy of the object with the new scope applied. For .gaSegmentFilters this can be either 'users' or 'sessions'. For metric expressions use either 'perUser', 'perSession', 'perHit' or 'perProduct'.

Value

The scope level as a character string, or returns a .gaSegmentFilter or gaMetExpr object with the newly set scope.

Methods (by class)

- object = gaSegMetExpr, value = missing: Return the scope of the supplied metric used within a segment definition.
- object = gaSegMetExpr, value = character: Set the scope, as described by a character value, to be applied to the supplied metric condition for use within a segment expression.
- object = gaMetExpr, value = character: Set the scope, as described by a character value, to be applied to the supplied metric condition for use within a segment expression.
- object = .gaSegmentFilter, value = missing: Returns the scope of the supplied .gaSegmentFilter.
- object = .gaSegmentFilter, value = character: Set the scope level of a .gaSegmentFilter to either "user" or "session" level.
- object = .gaSegmentFilter, value = character: Set the scope level of a .gaSegmentFilter to either "user" or "session" level.
- object = gaDynSegment, value = character: Set the scope level of a gaDynSegment to either "user" or "session" level.

See Also

Other dynamic segment functions: DynSegment, Exclude, Include, IsNegated, PerHit, PerProduct, PerSession, PerUser, SegmentConditionFilter, Segments, Segment

Other dynamic segment functions: DynSegment, Exclude, Include, IsNegated, PerHit, PerProduct, PerSession, PerUser, SegmentConditionFilter, Segments, Segment

Examples

```
sessions_with_value <- Expr(~eventValue > 0, metricScope = "perSession")
ScopeLevel(sessions_with_value)
users_with_value_sessions <- Include(sessions_with_value)
ScopeLevel(users_with_value_sessions) <- "users"
sessions_with_value_segment <- ScopeLevel(users_with_value_sessions, "sessions")
```

Segment *Segment*

Description

Define a segment for use in a query's segment list.

Usage

```
Segment(object, ...)

## S4 method for signature 'numeric'
Segment(object)

## S4 method for signature 'character'
Segment(object)

## S4 method for signature 'ANY'
Segment(object, ...)

## S4 method for signature '`NULL`'
Segment(object)

## S4 method for signature 'gaUserSegment'
Segment(object)
```

Arguments

object	A segment or other object that can be coerced into a segment, including dynamic segments, built-in and/or custom segments by their ID.
...	Other segment conditions, filters or filter lists to include in the segment's definition (ANDed)

Value

An object belonging to the .gaSegment superclass.

Methods (by class)

- numeric: Interpret the supplied numeric value as a segment ID.
- character: Interpret the supplied character value as a segment ID.
- ANY: Create a non-sequential segment using the supplied expressions.
- NULL: returns NULL
- gaUserSegment: Return the segment ID of the supplied GA Management API user segment.

See Also

Other dynamic segment functions: DynSegment, Exclude, Include, IsNegated, PerHit, PerProduct, PerSession, PerUser, ScopeLevel, SegmentConditionFilter, Segments

---

SegmentConditionFilter

SegmentConditionFilter

Description

Create a new gaSegmentConditionFilter object

Usage

```
SegmentConditionFilter(object, ..., negation, scope)

## S4 method for signature 'ANY'
SegmentConditionFilter(object, ..., negation, scope)
```

Arguments

object	An expression to be used as a non-sequential segment condition.
...	Other expressions to be Anded to the first expression provided.
negation	Optional logical TRUE or FALSE to match segments where this condition has not been met. Default is FALSE, i.e. inclusive filter.
scope	Optional scope, "users" or "sessions" (default).

Value

A gaSegmentConditionFilter object.

Methods (by class)

- ANY: Create a non-sequential segment condition filter from one or more expressions. All conditions within the filter must hold true within a single session if applied to a gaDynSegment scoped at session-level, or to a single hit if scoped at user-level.

See Also

Other dynamic segment functions: DynSegment, Exclude, Include, IsNegated, PerHit, PerProduct, PerSession, PerUser, ScopeLevel, Segments, Segment

Examples

```
bounced_sessions <- SegmentConditionFilter(Expr(~bounces > 0))
return_shoppers <- SegmentConditionFilter(
  Expr(~transactions > 1, metricScope = "perUser"),
  scope = "users"
)
```

Segments *Segments*

Description

Get the list of segments from the object or coerce the supplied objects into a a named list of segments.

Set the segments of the query object.

Usage

```
Segments(object, ...)

Segments(object) <- value

## S4 method for signature 'gaSegmentList'
Segments(object)

## S4 method for signature 'gaQuery'
Segments(object)

## S4 method for signature 'ANY'
Segments(object)

## S4 replacement method for signature 'gaQuery'
Segments(object) <- value
```

Arguments

object	A query object to get the segment list from or to set the segment list of.
...	Alternatively, provide one or more named arguments (segments or objects that can be coerced into segments) including dynamic segments, built-in and/or custom segments by their ID.
value	A named list of segments or a single segment.

Value

A gaSegmentList

Methods (by class)

- gaSegmentList: Returns itself
- gaQuery: Return the definition of the segment applied to the view.
- ANY: Coerce an object into a segmentList of length 1.
- gaQuery: Set the segments to be used within a query.

See Also

Other dynamic segment functions: DynSegment, Exclude, Include, IsNegated, PerHit, PerProduct, PerSession, PerUser, ScopeLevel, SegmentConditionFilter, Segment

Other dynamic segment functions: DynSegment, Exclude, Include, IsNegated, PerHit, PerProduct, PerSession, PerUser, ScopeLevel, SegmentConditionFilter, Segment

Examples

```
my_segments <- Segments(list(
  bounces = PerSession(Expr(~bounces != 0)),
  conversions = PerUser(Expr(~goalCompletionsAll > 0) | Expr(~transactions > 0)),
  mobile_or_tablet = Expr(~deviceCategory %in% c("mobile", "tablet")),
  multi_session_users = Include(PerUser(Expr(~sessions > 1)), scope = "users"),
  new_desktop_users = Expr(~deviceCategory == "desktop") & Expr(~userType == "new")
))

my_query <- GaQuery(view = "987654321")
my_segments_list <- list(
  bounces = PerSession(Expr(~bounces != 0)),
  conversions = PerUser(Expr(~goalCompletionsAll > 0) | Expr(~transactions > 0)),
  mobile_or_tablet = Expr(~deviceCategory %in% c("mobile", "tablet")),
  multi_session_users = Include(PerUser(Expr(~sessions > 1)), scope = "users"),
  new_desktop_users = Expr(~deviceCategory == "desktop") & Expr(~userType == "new")
)
Segments(my_query) <- my_segments_list
```

Sequence *Sequence*

Description

Create a new gaSequence object

Usage

```
Sequence(object, ..., negation, scope)

## S4 method for signature '.compoundExpr'
Later(object)

## S4 method for signature '.compoundExpr'
Then(object)

## S4 method for signature '.compoundExpr'
First(object)

## S4 method for signature 'ANY'
Sequence(object, ..., negation, scope)
```

Arguments

object	A sequence step or another expression that should be coerced to a sequence condition.
...	Other steps within the sequence condition, in the order in which they should be applied.
negation	Logical TRUE or FALSE to match segments where this sequence has not occurred.
scope	Optional scope, "users" or "sessions".

Methods (by class)

- .compoundExpr: Defines a sequence step using the supplied expression that does not need to be immediately at the start nor immediately following any preceding step.
- .compoundExpr: Defines a sequential step using the supplied expression that should immediately follow any preceding step or be the very first required interaction in any sequences being matched to this sequential segment definition.
- .compoundExpr: Alias to Then.
- ANY: Return a sequence of one or more steps using the supplied expression(s) that define the steps, where those step can occur anywhere within the sequences of interactions being matched, but in the order specified.

See Also

Other sequence segment functions: First, Later, Then

Examples

```
expr1 <- Expr(~pagepath == "/")
expr2 <- Expr(~eventCategory == "video")
expr3 <- Expr(~timeOnPage > 10)
expr4 <- Expr(~transactionRevenue > 10)
expr5 <- expr1 & expr2
expr6 <- Expr(~eventAction == "click")
expr7 <- Expr(~eventCategory == "video") & Expr(~eventAction == "play")
expr8 <- Expr(~source == "google")
Segment(
  PerUser(
    expr1,        # treat an expression as 'condition type segment filter' by default
    PerHit(expr3)
  ),
  Sequence(
    expr2,
    Then(expr4),  # 'then' means 'immediately followed by'.
    Later(expr5)  # 'later' means 'followed by', but not necessarily immediately.
  ),
  Sequence(
    First(expr6), # First expressly means 'first interaction' within the date range.
    Then(expr7),  # By default, treat an expression within a sequence as happening
    expr8        # at any point after any preceding steps in the sequence, i.e. 'later'.
  )
)
```

---

sequential_segment *sequential_segment.*

Description

Create a sequence using non-standard evaluation syntax.

Usage

```
sequential_segment(steps)
```

Arguments

steps	a list of expressions (of one or more conditions), each repeating a step in the sequence.

Details

Steps must be separated by commas (,). ... denotes zero or more interactions may precede the step that follows, otherwise without ... then the there must not be any interactions between the adjacent steps in order for the results to match the sequence.

Examples

```
a <- Expr(~pagePath == "/home")
b <- Expr(~eventCategory == "Video") &
  Expr(~eventAction == "Play")
c <- Expr(~medium == "email")
s <- sequential_segment(list( ..., a, ..., b, c ))
```

SortBy *SortBy*

Description

Get the sortBy order of the query.

Set the order of rows returned by Google Analytics.

Usage

```
SortBy(object, ..., desc = logical(0), type = c("VALUE", "DELTA",
  "SMART", "HISTOGRAM_BUCKET", "DIMENSION_AS_INTEGER")[0L])

SortBy(object) <- value

## S4 method for signature '.varList'
SortBy(object, desc, type)

## S4 method for signature '`NULL`'
SortBy(object)

## S4 method for signature 'character'
SortBy(object, ..., desc = logical(0),
  type = c("VALUE", "DELTA", "SMART", "HISTOGRAM_BUCKET",
  "DIMENSION_AS_INTEGER")[0L])

## S4 method for signature '.query'
SortBy(object, value)

## S4 replacement method for signature '.query'
SortBy(object) <- value
```

Arguments

object	A character vector or list of dimensions or metrics to sort by. If character, then prefixing the dimension name with a "+" means ascending order or "-" for descending order. By default metrics are sorted in descending order, while dimensions are by default in ascending order. Alternatively, supply a query object and replacement dimensions and metrics or sort by.
...	Further dimensions or metrics to sort by, or if object is a query then the replacement list of dimensions or metrics to sort by.
desc	A logical vector, same length as the resulting list of dimension or metric variables, indicating which columns of the resulting query response should be sorted in descending order.
type	A character vector, same length as the vector of variables to sort by, indicating the method of sorting to be applied to each variable. Available sort types are "VALUE", "DELTA", "SMART", "HISTOGRAM_BUCKET" or "DIMENSION_AS_INTEGER".
value	The replacement dimensions and metrics for the supplied object.

Methods (by class)

- .varList: Coerce a .varList object to a .sortBy child-class.
- NULL: Returns NULL
- character: Return a sortBy object given by the variables named within a character vector, optionally denoted with - or + to indicate descending or ascending sorting for each variable respectively in order of precedence.
- .query: Replace the sort by argument of a query.
- .query: Replace the sort by argument of a query.

See Also

Other query object functions: Dimensions, MaxResults, Metrics, SamplingLevel, TableFilter

Other query object functions: Dimensions, MaxResults, Metrics, SamplingLevel, TableFilter

---

SplitDateRange *SplitDateRange*

Description

Splits a gaDateRange object into N pieces. Useful for splitting a query into smaller chunks in order to overcome sampling.

Usage

```
SplitDateRange(dateRange, N = 0L)
```

Arguments

dateRange	the gaDateRange object to be split
N	the number of the separate date ranges to be split into; use 0 for single days.

See Also

Other date range functions: Cohort, DateRange, EndDate, StartDate

---

StartDate *StartDate*

Description

Get the start date.

Set the start date.

Usage

```
StartDate(object, value)

StartDate(object) <- value
```

Arguments

object	Object to get start date of.
value	Value to set start date of object to.

See Also

Other date range functions: Cohort, DateRange, EndDate, SplitDateRange

Other date range functions: Cohort, DateRange, EndDate, SplitDateRange

---

TableFilter *TableFilter*

Description

Get the filter.

Set the filter.

Usage

```
TableFilter(object, value)

TableFilter(object) <- value

## S4 method for signature 'ANY,missing'
TableFilter(object)

## S4 method for signature '.query,missing'
TableFilter(object)

## S4 method for signature '.query,ANY'
TableFilter(object, value)

## S4 replacement method for signature '.query'
TableFilter(object) <- value
```

Arguments

`object`	The object to be coerced to a TableFilter or the query object to apply a table filter to.
`value`	The replacement table filter where `object` is a query.

Methods (by class)

- `object = ANY, value = missing`: Coerce the given object into a table filter.
- `object = .query, value = missing`: Return the TableFilter that has been applied to the given query.
- `object = .query, value = ANY`: Method to replace the table filter of a query
- `.query`: Method to replace the table filter of a query

See Also

Other query object functions: `Dimensions`, `MaxResults`, `Metrics`, `SamplingLevel`, `SortBy`

Other query object functions: `Dimensions`, `MaxResults`, `Metrics`, `SamplingLevel`, `SortBy`

Then *Then*

Description

Treat a step within a sequence as happening immediately after any preceding steps in the sequence, i.e. 'immediately following'.

Usage

```
Then(object, ...)
```

Arguments

object	The expression that should **immediately** precede others in the sequence.
...	Any other expressions that should **immediately** follow the first one but before any others in the sequence.

Value

A gaSegmentSequenceStep object, with the immediate flag set.

See Also

Sequence

Other sequence segment functions: First, Later, Sequence

Examples

```
purchased_immediately_after <- Then(Expr(~transactions > 0))
```

ValidGaOperand *ValidGaOperand.*

Description

Checks whether an operand value is valid for a selected dimension.

Usage

```
ValidGaOperand(var, operand)
```

Arguments

var	selected dimension to check operand against
operand	the operand value to check

Var *Var*

Description

Google Analytics dimension and metric variables.

Var<- sets the value of an object belonging to the superclass .var or sets the var slot of an expression object belonging to superclass .expr

Usage

```
Var(object, ...)

Var(object) <- value

GaVar(object, ...)

GaVar(object) <- value

McfVar(object, ...)

McfVar(object) <- value

RtVar(object, ...)

RtVar(object) <- value

## S4 method for signature 'character'
Var(object)

## S4 replacement method for signature '.var,character'
Var(object) <- value

## S4 method for signature '.expr'
Var(object)

## S4 replacement method for signature '.expr,character'
Var(object) <- value

## S4 method for signature '.gaVarList'
Var(object)

## S4 method for signature 'character'
GaVar(object)

## S4 replacement method for signature '.gaVar,character'
GaVar(object) <- value
```

```
## S4 method for signature '.expr'
GaVar(object)

## S4 replacement method for signature '.expr,character'
GaVar(object) <- value

## S4 method for signature '.gaVarList'
GaVar(object)

## S4 method for signature 'ANY'
McfVar(object)

## S4 method for signature 'ANY'
RtVar(object)
```

Arguments

object	An object that inherits from or extends the class .var, including gaDimVar, gaMetVar, mcfDimVar, mcfMetVar, rtDimVar, rtMetVar, gaExpr, mcfExpr, rtExpr, gaDimensions, gaMetrics, mcfDimensions, mcfMetrics, rtDimensions and rtMetrics.
...	A replacement value for object coerced to class .var.
value	any object that can be coerced to a valid object class.

Details

Var returns a .var object which is valid Google Analytics dimension or metric for use with the core reporting, multi-channel-funnel reporting or real-time reporting API.

Use Var to lookup a dimension or metric from the Google Analytics core reporting, multi-channel-funnel reporting, or real-time reporting APIs, for use in defining expressions (of superclass .expr) or (to be implemented) variable lists (of superclass .varList) such as query dimensions, metrics or sortBy parameters.

Var accepts either a character, .var, or .expr object. A character object will be coerced to a .var object by looking for a matching dimension or metric from the Core Reporting, Multi-Channel Funnel Reporting, and Real-Time Reporting APIs. Providing an .expr object will return the dimension or metric used within that Google Analytics expression.

Value

An object inheriting from the superclass .var

Methods (by class)

- character: Coerce a character to '.var'.
- object = .var,value = character: Set a '.var' object to a new value coerced from character.
- .expr: Get the variable of an expression object.

- `object = .expr,value = character`: Set the variable of an expression object using a character value to be coerced to '.var'.
- `.gaVarList`: Get the variables within a variable list object, such as sortBy, dimensions or metrics.
- `character`: GaVar takes a GA variable name and determines whether to return a Dimension or Metric object
- `object = .gaVar,value = character`: Set the Var of a gaExpr object.
- `.expr`: Get the variable from expression object coerced to '.garVar'.
- `object = .expr,value = character`: Set the variable of an expression to a .gaVar as named by a character value.
- `.gaVarList`: Get the variables of a .gaVarList.
- `ANY`: McfVar takes a MCF variable and determines whether to return a Dimension or Metric object
- `ANY`: McfVar takes a RT variable and determines whether to return a Dimension or Metric object

See Also

- Core Reporting API dimensions and metrics
- Multi-Channel-Funnel Reporting API dimensions and metrics
- Real-Time Reporting API dimensions and metrics

Examples

```
Var("source")
dim <- Var("ga:medium")
Var(dim)
paid_traffic <- Expr(dim, "==", "cpc")
Var(paid_traffic)

expr1 <- Expr("pageviews", '>', 10)
Var(expr1) <- "uniquePageviews"
```

`xor` *xor*

Description

`xor` produces a compound expression that gives the EXCLUSIVE-OR of two expressions.

Usage

```
xor(x, y)

## S4 method for signature '.compoundExpr,.compoundExpr'
xor(x, y)
```

Arguments

x, y	Conditions for an EXCLUSIVE-OR expression.

Methods (by class)

- x = .compoundExpr,y = .compoundExpr: Exclusive-OR of two expressions.

See Also

Other boolean functions: And, Not, Or

Examples

```
either_enquired_or_downloaded <- xor(
  Expr(~eventCategory == "enquiry"),
  Expr(~eventCategory == "download")
)
```

Index