Package ‘ganalytics’

March 2, 2019

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, htr (>= 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,
tidyrr, spelling, covr

Collate 'Bool-generics.R' 'utilis.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'
'Comparatordgenerics.R' 'comparator-coerce.R'
'Comparatordmethods.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'
R topics documented:

'expr-coerce.R' 'Expr-methods.R' 'GaGetCoreReport.R'
'GaView-generic.R' 'view-coerce.R' 'GaView-methods.R'
'ga-api-coerce.R' 'GetGa-methods.R' 'Query-methods.R'
'segment-coerce.R' 'Sequence-generic.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 <johannedeboer@gmail.com>
Repository CRAN
Date/Publication 2019-03-02 09:20:03 UTC

R topics documented:

And ................................................................. 3
Cohort .............................................................. 4
Comparator ......................................................... 5
comparators ....................................................... 6
DateRange .......................................................... 9
Dimensions ......................................................... 11
DynSegment ......................................................... 12
EndDate ............................................................. 13
Exclude ............................................................. 14
Expr ................................................................. 15
First ................................................................. 16
GaAccount .......................................................... 17
GaAccounts ........................................................ 17
GaAccountSummaries ............................................. 17
GaAccountSummary .............................................. 18
GaCreds .............................................................. 18
GaExpr .............................................................. 19
GaMetaUpdate ...................................................... 20
GaQuery ............................................................ 20
GaUserSegment .................................................... 21
GaUserSegments .................................................. 22
ga_view_selector .................................................. 22
GetGaData .......................................................... 24
GetGaData, query-method ......................................... 24
**Description**

Logical AND of two or more expressions.
Usage

\[
\text{And}(\text{object}, \ldots)
\]

\[
\text{## S4 method for signature } '.compoundExpr' \\
\text{And}(\text{object}, \ldots)
\]

\[
\text{## S4 method for signature } '.compoundExpr,.compoundExpr' \\
e1 \& e2
\]

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>An object to include within the ANDed expression.</td>
</tr>
<tr>
<td>\ldots</td>
<td>Additional objects to include within the ANDed expression.</td>
</tr>
<tr>
<td>e1</td>
<td>first expression</td>
</tr>
<tr>
<td>e2</td>
<td>second expression</td>
</tr>
</tbody>
</table>

Value

An object of class andExpr

Methods (by class)

- `.compoundExpr`: Logical-AND of two or more expressions.

See Also

Other boolean functions: `Not, Or, xor`

Examples

\[
purchased_on_mobile \leftarrow \text{Expr}(-\text{deviceCategory} == \text{"mobile"}) \& \text{Expr}(-\text{transactions} > 0)
\]
## 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`

### Arguments

- **object**: The object to be coerced to a `.comparator` subclass or the expression object of which to obtain its comparator.
- **value**: Used by certain methods.
- **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`
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 %endsWith% 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'

```r
e1 != e2
```

## S4 method for signature '.var,.metOperand'

```r
e1 > e2
```

## S4 method for signature '.var,.metOperand'

```r
e1 < e2
```

## S4 method for signature '.var,.metOperand'

```r
e1 >= e2
```

## S4 method for signature '.var,.metOperand'

```r
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)

---

**Description**

Get the date range.
Set the date range.

**Usage**

`DateRange(object, endDate)`

`DateRange(object) <- value`

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

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

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

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

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

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

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

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

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

```r
## 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.
dimensions

- 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**

- **Description**
  - Get the dimensions of the object.
  - Set the dimensions for the object.

- **Usage**
  - Dimensions(object, ...)
  - Dimensions(object) <- value

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

## 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

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

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

```r
## 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))
)
```

Description

Get the end date of the date range.
Set the endDate of the date range.
Exclude

Usage

```r
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`

Description

Set the negation flag of a segment filter to TRUE.

Usage

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

**Examples**

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

**Description**

Define a Google Analytics expression.

**Usage**

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

### 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.

**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".
See Also

Other expression generators: GaExpr, McfExpr, RtExpr

Examples

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

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

```r
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

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

**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

**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

**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
GaExpr

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.

---

Description

Create a Core Reporting API expression.

Usage

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

```r
## 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. 
jbounces <- GaExpr("bounces", ">", 0)
See Also

Other expression generators: `Expr`, `McfExpr`, `RtExpr`

Examples

```r
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**

```r
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**

```r
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

Description

Get a user or system defined segment.

Usage

```r
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.

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)
ga_view_selector

```r
## 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.
- `object`: Replaces the view being used by a query.

**Examples**

```r
## 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

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.

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)

```r
## 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`
Description
Tests whether a segment filter is negated.
Sets the negation flag of a segment filter.

Usage

\texttt{IsNegated(object)}

\texttt{IsNegated(object) <- value}

\texttt{## S4 method for signature '.\texttt{gaSegmentFilter}'}
\texttt{IsNegated(object)}

\texttt{## S4 replacement method for signature '.\texttt{gaSegmentFilter,logical}'}
\texttt{IsNegated(object) <- value}

Arguments

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

Methods (by class)

- \texttt{.\texttt{gaSegmentFilter}}: Tests whether a segment filter is negated, i.e. used to define an exclude filter for the segment.
- \texttt{object = .\texttt{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: \texttt{DynSegment, Exclude, Include, PerHit, PerProduct, PerSession, PerUser, ScopeLevel, SegmentConditionFilter, Segments, Segment}
IsRegEx

Examples

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

Description

Checks for a regular expression.

Usage

```r
IsRegEx(object)
```

## S4 method for signature '.dimComparator'

```r
IsRegEx(object)
```

## S4 method for signature '.expr'

```r
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**

**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**

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

---

**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

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

Description

Create a Multi-Channel Funnel Reporting API query object

Usage

```r
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

---

### Description

Get the metrics of the object.
Set the metrics of the object.

### Usage

```r
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
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)

<table>
<thead>
<tr>
<th>Operand</th>
<th>Operand</th>
</tr>
</thead>
</table>

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.
**Description**

Logical OR of two or more expressions.

**Usage**

```
Or(object, ...)
```

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

```r
## 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 expressions 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)
```
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

```r
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(~eventCategory == "Video" & ~eventAction == "Play" & ~eventLabel == "Intro")

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

\[
\text{with\_products\_added\_more\_than\_once} \leftarrow \text{PerProduct}(\neg \text{product\_adds\_to\_cart} > 1))
\]

Description

Set the scope of a {\@\text{.\text{gaSegmentFilter}}} or {\@\text{\text{gaMetExpr}}} object to session-level.

Usage

\begin{verbatim}
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)
\end{verbatim}

Arguments

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

Value

A {\@\text{\text{gaMetExpr}}}..{\@\text{\text{.\text{gaSegmentFilter}}}} or {\@\text{\text{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**

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

---

**PerUser**

**Description**

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

**Usage**

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

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

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

```r
# 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

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

Create a Real-Time Reporting API expression.

**Usage**

```r
RtExpr(object, comparator, operand)
```

```r
## 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**

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

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

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)
## ScopeLevel

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

### 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

```r
ScopeLevel

## Description

Get or set the scope level of a .gaSegmentFilter or 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)
```
ScopeLevel

```r
## 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.
Segment

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")

<table>
<thead>
<tr>
<th>Segment</th>
<th>Segment</th>
</tr>
</thead>
</table>

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)
SegmentConditionFilter

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

Description
Create a new gaSegmentConditionFilter object

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

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>An expression to be used as a non-sequential segment condition.</td>
</tr>
<tr>
<td>...</td>
<td>Other expressions to be Anded to the first expression provided.</td>
</tr>
<tr>
<td>negation</td>
<td>Optional logical TRUE or FALSE to match segments where this condition has not been met. Default is FALSE, i.e. inclusive filter.</td>
</tr>
<tr>
<td>scope</td>
<td>Optional scope, &quot;users&quot; or &quot;sessions&quot; (default).</td>
</tr>
</tbody>
</table>

Value
A gaSegmentConditionFilter object.
Segments

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

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

Description

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

Usage

```r
Segments(object, ...)

Segments(object) <- value
```

## S4 method for signature 'gaSegmentList'

```r
Segments(object)
```

## S4 method for signature 'gaQuery'

```r
Segments(object)
```

## S4 method for signature 'ANY'

```r
Segments(object)
```

## S4 replacement method for signature 'gaQuery'

```r
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

```r
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
```
**Description**

Create a new gaSequence object

**Usage**

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

```r
## S4 method for signature 'compoundExpr'
Later(object)
```

```r
## S4 method for signature 'compoundExpr'
Then(object)
```

```r
## S4 method for signature 'compoundExpr'
First(object)
```

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

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>object</td>
<td>A sequence step or another expression that should be coerced to a sequence condition.</td>
</tr>
<tr>
<td>...</td>
<td>Other steps within the sequence condition, in the order in which they should be applied.</td>
</tr>
<tr>
<td>negation</td>
<td>Logical TRUE or FALSE to match segments where this sequence has not occurred.</td>
</tr>
<tr>
<td>scope</td>
<td>Optional scope, &quot;users&quot; or &quot;sessions&quot;.</td>
</tr>
</tbody>
</table>

**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

```r
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'.
  )
)
```

Description

Create a sequence using non-standard evaluation syntax.

Usage

```r
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

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

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

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

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

Description

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

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))
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'

```r
eGaVar(object)
```

## S4 replacement method for signature '.expr,character'

```r
eGaVar(object) <- value
```

## S4 method for signature '.gaVarList'

```r
eGaVar(object)
```

## S4 method for signature 'ANY'

```r
eMcfVar(object)
```

## S4 method for signature 'ANY'

```r
eRtVar(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`.

- **value**: A replacement value for object coerced to class `.var`.

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

Usage

```r
xor(x, y)
```

## S4 method for signature '.compoundExpr,.compoundExpr'

```r
xor(x, y)
```
Arguments

\( x, \ y \) Conditions for an EXCLUSIVE-OR expression.

Methods (by class)

- \( x = .\text{compoundExpr}, y = .\text{compoundExpr} \): Exclusive-OR of two expressions.

See Also

Other boolean functions: \texttt{And}, \texttt{Not}, \texttt{Or}

Examples

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

!., comparator-method (Not), 33
!., expr-method (Not), 33
!., gaSegmentFilter-method (Not), 33
!., orExpr-method (Not), 33
!=., var., operand-method (comparators), 6
<., var., metOperand-method (comparators), 6
<=., var., metOperand-method (comparators), 6
==., var., operand-method (comparators), 6
>, var., metOperand-method (comparators), 6
>=., var., metOperand-method (comparators), 6
%between%, (comparators), 6
%between%, var., operand-method (comparators), 6
%contains%, (comparators), 6
%contains%, var., dimOperand-method (comparators), 6
%ends_with%, (comparators), 6
%ends_with%, var., dimOperand-method (comparators), 6
%in%, (comparators), 6
%in%, var., operand-method (comparators), 6
%matches%, (comparators), 6
%matches%, var., dimOperand-method (comparators), 6
%starts_with%, (comparators), 6
%starts_with%, var., dimOperand-method (comparators), 6
&., compoundExpr-method (And), 3
'!' (Not), 33
'&' (And), 3

And, 3, 34, 35, 59
And., compoundExpr-method (And), 3

Cohort, 4, 11, 14, 53
Cohort<- (Cohort), 4
Comparator, 5, 8
Comparator.<- (Comparator), 5
Comparator<- (Comparator), 5
Comparators, 6, 6

DateRange, 5, 9, 14, 53
DateRange., standardQuery, missing-method (DateRange), 9
DateRange, ANY, ANY-method (DateRange), 9
DateRange, ANY, missing-method (DateRange), 9
DateRange, gaView, ANY-method (DateRange), 9
DateRange<- (DateRange), 9
DateRange<-, .standardQuery-method (DateRange), 9
DateRange<-, ANY-method (DateRange), 9
Dimensions, 11, 30, 32, 43, 52, 54
Dimensions, .query-method (Dimensions), 11
Dimensions, ANY-method (Dimensions), 11
Dimensions<- (Dimensions), 11
Dimensions<-, .query-method (Dimensions), 11
DynSegment, 12, 14, 26, 27, 36, 37, 39, 40, 45–48
DynSegment, ANY-method (DynSegment), 12
DynSegment, gaDynSegment-method (DynSegment), 12

EndDate, 5, 11, 13, 53
EndDate., standardQuery-method (DateRange), 9
EndDate, character-method (DateRange), 9
EndDate, dateRange-method (DateRange), 9
EndDate, gaView-method (DateRange), 9
EndDate, Interval-method (DateRange), 9
EndDate<- (EndDate), 13
EndDate<-.standardQuery-method
(DateRange), 9
EndDate<-.dateRange-method (DateRange), 9
Exclude, 13, 14, 26, 27, 36, 37, 39, 40, 45–48
Exclude, ANY-method (Exclude), 14
Expr. 15, 20, 31, 41
Expr., expr, ANY-method (Expr), 15
Expr, character, character-method (Expr), 15
Expr, formula, ANY-method (Expr), 15
First, 16, 29, 50, 55
First, .compoundExpr-method (Sequence), 49

ga_view_selector, 22
GaAccount, 17
GaAccounts, 17
GaAccountSummaries, 17
GaAccountSummary, 18
GaCreds, 18
GaCreds, .query, ANY-method (GaCreds), 18
GaCreds, .query, list-method (GaCreds), 18
GaCreds, character, ANY-method (GaCreds), 18
GaCreds, missing, ANY-method (GaCreds), 18
GaCreds<- (GaCreds), 18
GaCreds<-., query, list-method (GaCreds), 18
GaExpr, 16, 19, 31, 41
GaExpr, character, character-method
(GaExpr), 19
GaMetaUpdate, 20
GaQuery, 20
GaUserSegment, 21
GaUserSegments, 22
GaVar (Var), 56
GaVar, .expr-method (Var), 56
GaVar, .gaVarList-method (Var), 56
GaVar, character-method (Var), 56
GaVar<- (Var), 56
GaVar<-, .expr, character-method (Var), 56
GaVar<-, .gaVar, character-method (Var), 56
GaView (ga_view_selector), 22
GaView, .query, ANY-method
(ga_view_selector), 22
GaView, .query, missing-method
(ga_view_selector), 22
GaView, ANY, missing-method
(ga_view_selector), 22
GaView, gaAccount, missing-method
(ga_view_selector), 22
GaView, gaProperty, missing-method
(ga_view_selector), 22
GaView<.< (ga_view_selector), 22
GaView<-.query-method
(ga_view_selector), 22
GetGaData, 24
GetGaData, .query-method, 24
GoogleApiCreds, 25
GtmAccount, 25
GtmAccounts, 26
Include, 13, 14, 26, 27, 36, 37, 39, 40, 45–48
Include, ANY-method (Include), 26
IsNegated, 13, 14, 26, 27, 36, 37, 39, 40, 45–48
IsNegated, .gaSegmentFilter-method
(IsNegated), 27
IsNegated<.< (IsNegated), 27
IsNegated<-.gaSegmentFilter, logical-method
(IsNegated), 27
IsRegEx, 28
IsRegEx, .dimComparator-method
(IsRegEx), 28
IsRegEx, .expr-method (IsRegEx), 28
Later, 16, 29, 50, 55
Later, .compoundExpr-method (Sequence), 49

MaxResults, 12, 29, 32, 43, 52, 54
MaxResults, .query, ANY-method
(MaxResults), 29
MaxResults, .query, missing-method
(MaxResults), 29
MaxResults<.< (MaxResults), 29
MaxResults<-.query-method
(MaxResults), 29
McfExpr, 16, 20, 30, 41
McfExpr, character, character-method
(McfExpr), 30
McfQuery, 31
INDEX

McfVar (Var), 56
McfVar, ANY-method (Var), 56
McfVar<- (Var), 56
Metrics, 12, 30, 32, 43, 52, 54
Metrics, .query-method (Metrics), 32
Metrics, ANY-method (Metrics), 32
Metrics<-(Metrics), 32
Metrics<-.query-method (Metrics), 32

Not, 4, 33, 35, 59
Not,.comparator-method (Not), 33
Not,.expr-method (Not), 33
Not,.gaSegmentFilter-method (Not), 33
Not,.orExpr-method (Not), 33

Operand, 34
Operand,.expr-method (Operand), 34
Operand<-(Operand), 34
Operand<-.expr-method (Operand), 34
Or, 4, 34, 35, 59
Or,.compoundExpr-method (Or), 35

PerHit, 13, 14, 26, 27, 36, 37, 39, 40, 45–48
PerHit,.compoundExpr-method (PerHit), 36
PerHit,.formula-method (PerHit), 36
PerHit,.gaMetExpr-method (PerHit), 36
PerProduct, 13, 14, 26, 27, 36, 37, 39, 40, 45–48
PerProduct,.formula-method (PerProduct), 37
PerProduct,.gaMetExpr-method (PerProduct), 37
PerSession, 13, 14, 26, 27, 36, 37, 38, 40, 45–48
PerSession,.ANY-method (PerSession), 38
PerSession,.formula-method (PerSession), 38
PerSession,.gaMetExpr-method (PerSession), 38
PerSession,.gaSegmentSequenceFilter-method (PerSession), 38
PerSession,.gaSegmentSequenceStep-method (PerSession), 38
PerUser, 13, 14, 26, 27, 36, 37, 39, 45–48
PerUser,.ANY-method (PerUser), 39
PerUser,.formula-method (PerUser), 39
PerUser,.gaMetExpr-method (PerUser), 39
PerUser,.gaSegmentSequenceFilter-method (PerUser), 39
PerUser,.gaSegmentSequenceStep-method (PerUser), 39
RtExpr, 16, 20, 31, 41
RtExpr,.character,.character-method (RtExpr), 41
RtQuery, 42
RtVar (Var), 56
RtVar,.ANY-method (Var), 56
RtVar<- (Var), 56

SamplingLevel, 12, 30, 32, 42, 52, 54
SamplingLevel,.standardQuery,.ANY-method (SamplingLevel), 42
SamplingLevel,.standardQuery,.missing-method (SamplingLevel), 42
SamplingLevel,.data.frame,.ANY-method (SamplingLevel), 42
SamplingLevel,.data.frame,.missing-method (SamplingLevel), 42
SamplingLevel<- (SamplingLevel), 42
ScopeLevel, 13, 14, 26, 27, 36, 37, 39, 40, 43, 46–48
ScopeLevel,.gaSegmentFilter,.character-method (ScopeLevel), 43
ScopeLevel,.gaSegmentFilter,.missing-method (ScopeLevel), 43
ScopeLevel,.gaSegmentSequenceStep,.character-method (ScopeLevel), 43
ScopeLevel,.gaSegmentSequenceStep,.missing-method (ScopeLevel), 43
ScopeLevel<- (ScopeLevel), 43
ScopeLevel<-,.gaSegmentFilter,.character-method (ScopeLevel), 43
ScopeLevel<-,.gaSegmentSequenceStep,.character-method (ScopeLevel), 43
ScopeLevel<-,.gaMetExpr,.character-method (ScopeLevel), 43
ScopeLevel<-,.gaMetExpr,.missing-method (ScopeLevel), 43
Segment, 13, 14, 26, 27, 36, 37, 39, 40, 45, 47, 48
Segment,.ANY-method (Segment), 45
Segment,.character-method (Segment), 45
Segment,.gaUserSegment-method (Segment), 45
Segment,.NULL-method (Segment), 45
Segment,.numeric-method (Segment), 45
SegmentConditionFilter, 13, 14, 26, 27, 36, 37, 39, 40, 45, 46, 48
INDEX

SegmentConditionFilter, ANY-method (SegmentConditionFilter), 46
Segments, 13, 14, 26, 27, 36, 37, 39, 40, 45–47, 47
Segments, ANY-method (Segments), 47
Segments, gaQuery-method (Segments), 47
Segments, gaSegmentList-method (Segments), 47
Segments<- (Segments), 47
Segments<-, gaQuery-method (Segments), 47
Sequence, 16, 29, 49, 55
Sequence, ANY-method (Sequence), 49
sequential_segment, 50
SortBy, 12, 30, 32, 43, 51, 54
SortBy, query-method (SortBy), 51
SortBy, varList-method (SortBy), 51
SortBy, character-method (SortBy), 51
SortBy, NULL-method (SortBy), 51
SortBy<- (SortBy), 51
SortBy<-, query-method (SortBy), 51
SplitDateRange, 5, 11, 14, 52, 53
StartDate, 5, 11, 14, 53, 53
StartDate, standardQuery-method (DateRange), 9
StartDate, character-method (DateRange), 9
StartDate, dateRange-method (DateRange), 9
StartDate, gaView-method (DateRange), 9
StartDate, Interval-method (DateRange), 9
StartDate<- (StartDate), 53
StartDate<-, standardQuery-method (DateRange), 9
StartDate<-, dateRange-method (DateRange), 9
TableFilter, 12, 30, 32, 43, 52, 53
TableFilter, query, ANY-method (TableFilter), 53
TableFilter, query, missing-method (TableFilter), 53
TableFilter, ANY, missing-method (TableFilter), 53
TableFilter<- (TableFilter), 53
TableFilter<-, query-method (TableFilter), 53
Then, 16, 29, 50, 54
Then, compoundExpr-method (Sequence), 49
tibble, 24

ValidGaOperand, 55
Var, 7, 56
Var, expr-method (Var), 56
Var, gaVarList-method (Var), 56
Var, character-method (Var), 56
Var<- (Var), 56
Var<-, expr, character-method (Var), 56
Var<-, var, character-method (Var), 56
xor, 4, 34, 35, 58
xor, compoundExpr, compoundExpr-method (xor), 58