Package 'FireBrowseR'

April 8, 2015

Type Package

Title An API client for Broads Firehose Pipeline.

Version 0.1

Date 2015-03-11

Author Mario Deng

Maintainer Mario Deng<mariodeng@googlemail.com>

Description Get direct and fast informations from massive genomic data sets provided by Broads Firehose Pipeline / The Cancer Genome Atlas.

License MIT

LazyData TRUE

Depends R(>= 2.10.0)

Suggests testthat, jsonlite

R topics documented:

`Analyses.CopyNumber.Genes.All`

Retrieve all data by genes Gistic2 results.

Description

This service provides access to the Gistic2 all_data_by_genes.txt output data. This data is a gene-level table of copy number values for all samples. The returned copy number values are in units (copy number - 2) so that no amplification or deletion is 0, genes with amplifications have positive values, and genes with deletions are negative values. The data are converted from marker level to gene level using the extreme method: a gene is assigned the greatest amplification or the least deletion value among the markers it covers. Results may be filtered by cohort, gene, or barcode, but at least one gene OR barcode must be supplied.

Usage

```
Analyses.CopyNumber.Genes.All(format = "csv", cohort = "", gene = "",
  tcga_participant_barcode = "", page = 1, page_size = 250,
  sort_by = "cohort")
```

Arguments

`format`	A string identifying the data type returned. Using json requires the jsonlite package. `json`, `csv` or `tsv` are available. In this package `tsv` and `csv` are identical.
`cohort`	A character vector indicating the cohort to query, empty string queries all cohorts. See `Metadata.Cohorts` for available cohorts.
`gene`	A character vector of gene symbols. At least one gene OR Barcode is required.
`tcga_participant_barcode`	A character vector containing TCGA Barcodes. Empty string returns all patients. Either one gene OR barcode is required.
`page`	Subset to be returned.
`page_size`	Number of records per page, max. is 2000.
`sort_by`	character indicating the to column which is used for sorting.

Value

A `list`, if format is `json`, elsewise a `data.frame`

Examples

```
format = "csv"
cohort = "PRAD"
gene = "PTEN"
tcga_participant_barcode = c("TCGA-J4-A67O", "TCGA-G9-6496")
page = 1
page_size = 250
sort_by = "cohort"

obj = Analyses.CopyNumber.Genes.All(format = format,
                                    cohort = cohort,
                                    gene = gene,
                                    tcga_participant_barcode = tcga_participant_barcode,
                                    page = page,
                                    page_size = page_size,
                                    sort_by = sort_by)

gene = ""
obj = Analyses.CopyNumber.Genes.All(format = format,
                                    cohort = cohort,
                                    gene = gene,
                                    tcga_participant_barcode = tcga_participant_barcode,
                                    page = page,
                                    page_size = page_size,
                                    sort_by = sort_by)

format = "json"
obj = Analyses.CopyNumber.Genes.All(format = format,
                                    cohort = cohort,
                                    gene = gene,
                                    tcga_participant_barcode = tcga_participant_barcode,
                                    page = page,
                                    page_size = page_size,
                                    sort_by = sort_by)
```

Analyses.CopyNumber.Genes.Amplified

Retrieve Gistic2 significantly amplified genes results.

Description

This service provides access to the Gistic2 amp_genes.conf_99.txt output data.

Usage

```
Analyses.CopyNumber.Genes.Amplified(format = "csv", cohort = "",
  gene = "", q = "", page = 1, page_size = 250, sort_by = "cohort")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.

cohort	A character vector indicating the cohort to query, empty string queries all cohorts. See Metadata.Cohorts for available cohorts.
gene	A character vector of gene symbols. At least one gene OR Barcode is required.
q	Only return results with Q-value <= given threshold. For details please see https://www.broadinstitute.org/cancer/cga/mutsig
page	Subset to be returned.
page_size	Number of records per page, max. is 2000.
sort_by	character indicating the to column which is used for sorting.

Value

A list, if format is json, elsewise a data.frame

Examples

```
format = "csv"
cohort = ""
gene = "PTEN"
q = 0.05
page = 1
page_size = 250
sort_by = "cohort"

obj = Analyses.CopyNumber.Genes.Amplified(format = format,
                                          cohort = cohort,
                                          gene = gene,
                                          q = q,
                                          page = page,
                                          page_size = page_size,
                                          sort_by = sort_by)

gene = "PFN3"
obj = Analyses.CopyNumber.Genes.Amplified(format = format,
                                          cohort = cohort,
                                          gene = gene,
                                          q = q,
                                          page = page,
                                          page_size = page_size,
                                          sort_by = sort_by)

format = "json"
obj = Analyses.CopyNumber.Genes.Amplified(format = format,
                                          cohort = cohort,
                                          gene = gene,
                                          q = q,
                                          page = page,
                                          page_size = page_size,
                                          sort_by = sort_by)
```

`Analyses.CopyNumber.Genes.Deleted` *Retrieve Gistic2 significantly deleted genes results.*

Description

This service provides access to the Gistic2 del_genes.conf_99.txt output data.

Usage

```
Analyses.CopyNumber.Genes.Deleted(format = "csv", cohort = "", gene = "",
  q = "", page = 1, page_size = 250, sort_by = "cohort")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.
cohort	A character vector indicating the cohort to query, empty string queries all cohorts. See Metadata.Cohorts for available cohorts.
gene	A character vector of gene symbols. At least one gene OR Barcode is required.
q	Only return results with Q-value <= given threshold. For details please see https://www.broadinstitute.org/cancer/cga/mutsig
page	Subset to be returned.
page_size	Number of records per page, max. is 2000.
sort_by	character indicating the to column which is used for sorting.

Value

A list, if format is json, elsewise a data.frame

Examples

```
format = "csv"
cohort = ""
gene = "PTEN"
q = 0.05
page = 1
page_size = 250
sort_by = "cohort"

obj = Analyses.CopyNumber.Genes.Deleted(format = format,
                                        cohort = cohort,
                                        gene = gene,
                                        q = q,
                                        page = page,
                                        page_size = page_size,
                                        sort_by = sort_by)

gene = "PFN3"
obj = Analyses.CopyNumber.Genes.Deleted(format = format,
                                        cohort = cohort,
```

```
                                                gene = gene,
                                                q = q,
                                                page = page,
                                                page_size = page_size,
                                                sort_by = sort_by)

format = "json"
obj = Analyses.CopyNumber.Genes.Deleted(format = format,
                                                cohort = cohort,
                                                gene = gene,
                                                q = q,
                                                page = page,
                                                page_size = page_size,
                                                sort_by = sort_by)
```

`Analyses.CopyNumber.Genes.Focal`

Retrieve focal data by genes Gistic2 results.

Description

This service provides access to the Gistic2 focal_data_by_genes.txt output data. This outpuat is similar to the all_data_by_genes.txt output, but using only focal events with lengths greater than the focal length cutoff. This data is a gene-level table of copy number values for all samples. The returned copy number values are in units (copy number - 2) so that no amplification or deletion is 0, genes with amplifications have positive values, and genes with deletions are negative values. The data are converted from marker level to gene level using the extreme method: a gene is assigned the greatest amplification or the least deletion value among the markers it covers. Results may be filtered by cohort, gene, and/or barcode, but at least one gene OR barcode must be supplied.

Usage

```
Analyses.CopyNumber.Genes.Focal(format = "csv", cohort = "", gene = "",
  tcga_participant_barcode = "", page = 1, page_size = 250,
  sort_by = "cohort")
```

Arguments

`format`	A string identifying the data type returned. Using json requires the jsonlite package. `json`, `csv` or `tsv` are available. In this package `tsv` and `csv` are identical.
`cohort`	A character vector indicating the cohort to query, empty string queries all cohorts. See `Metadata.Cohorts` for available cohorts.
`gene`	A character vector of gene symbols. At least one gene OR Barcode is required.
`tcga_participant_barcode`	A character vector containing TCGA Barcodes. Empty string returns all patients. Either one gene OR barcode is required.
`page`	Subset to be returned.
`page_size`	Number of records per page, max. is 2000.
`sort_by`	character indicating the to column which is used for sorting.

Value

A `list`, if format is `json`, elsewise a `data.frame`

Examples

```
format = "csv"
cohort = "PRAD"
gene = "PTEN"
tcga_participant_barcode = c("TCGA-J4-A67O", "TCGA-G9-6496")
page = 1
page_size = 250
sort_by = "cohort"

obj = Analyses.CopyNumber.Genes.Focal(format = format,
                                      cohort = cohort,
                                      gene = gene,
                                      tcga_participant_barcode = tcga_participant_barcode,
                                      page = page,
                                      page_size = page_size,
                                      sort_by = sort_by)

gene = ""
obj = Analyses.CopyNumber.Genes.Focal(format = format,
                                      cohort = cohort,
                                      gene = gene,
                                      tcga_participant_barcode = tcga_participant_barcode,
                                      page = page,
                                      page_size = page_size,
                                      sort_by = sort_by)

format = "json"
obj = Analyses.CopyNumber.Genes.Focal(format = format,
                                      cohort = cohort,
                                      gene = gene,
                                      tcga_participant_barcode = tcga_participant_barcode,
                                      page = page,
                                      page_size = page_size,
                                      sort_by = sort_by)
```

`Analyses.CopyNumber.Genes.Thresholded` *Retrieve all thresholded by genes Gistic2 results.*

Description

This service provides access to the Gistic2 all_thresholded_by_genes.txt output data. A gene-level table of discrete amplification and deletion indicators for all samples. A table value of 0 means no amplification or deletion above the threshold. Amplifications are positive numbers: 1 means amplification above the amplification threshold; 2 means amplifications larger to the arm level amplifications observed for the sample. Deletions are represented by negative table values: -1 represents deletion beyond the threshold; -2 means deletions greater than the minimum arm-level deletion observed for the sample. Results maybe filtered by cohort, gene or barcode, but at least one gene OR barcode must be supplied.

Usage

```
Analyses.CopyNumber.Genes.Thresholded(format = "csv", cohort = "",
  gene = "", tcga_participant_barcode = "", page = 1, page_size = 250,
  sort_by = "cohort")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.
cohort	A character vector indicating the cohort to query, empty string queries all cohorts. See Metadata.Cohorts for available cohorts.
gene	A character vector of gene symbols. At least one gene OR Barcode is required.
tcga_participant_barcode	A character vector containing TCGA Barcodes. Empty string returns all patients. Either one gene OR barcode is required.
page	Subset to be returned.
page_size	Number of records per page, max. is 2000.
sort_by	character indicating the to column which is used for sorting.

Value

A list, if format is json, elsewise a data.frame

Examples

```
format = "csv"
cohort = "PRAD"
gene = "PTEN"
tcga_participant_barcode = c("TCGA-J4-A67O", "TCGA-G9-6496")
page = 1
page_size = 250
sort_by = "cohort"

obj = Analyses.CopyNumber.Genes.Thresholded(format = format,
                                cohort = cohort,
                                gene = gene,
                                tcga_participant_barcode = tcga_participant_barcode,
                                page = page,
                                page_size = page_size,
                                sort_by = sort_by)

gene = ""
obj = Analyses.CopyNumber.Genes.Thresholded(format = format,
                                cohort = cohort,
                                gene = gene,
                                tcga_participant_barcode = tcga_participant_barcode,
                                page = page,
                                page_size = page_size,
                                sort_by = sort_by)

format = "json"
obj = Analyses.CopyNumber.Genes.Thresholded(format = format,
                                cohort = cohort,
```

```
                                gene = gene,
                                tcga_participant_barcode = tcga_participant_barcode,
                                page = page,
                                page_size = page_size,
                                sort_by = sort_by)
```

`Analyses.FeatureTable` *Retrieve aggregated analysis features table.*

Description

This service returns part or all of the so-called feature table; which aggregates the most important findings across ALL pipelines in the GDAC Firehose analysis workflow into a single table for simple access. For more details please visit the online documentation (https://confluence.broadinstitute.org/display/GDAC/Documentation/#Documentation-FeatureTable). Please note that the service is still undergoing experimental evaluation and does not return JSON format.

Usage

```
Analyses.FeatureTable(format = "csv", cohort = "", column = "",
  page = 1, page_size = 250)
```

Arguments

`format`	Either `tsv` or `csv`, here `json` is not possible, but comming soon.
`cohort`	A character vector indicating the cohort to query, empty string queries all cohorts. See `Metadata.Cohorts` for available cohorts.
`column`	Comma separated list of which data columns/fields to return.
`page`	Subset to be returned.
`page_size`	Number of records per page, max. is 2000.

Value

A `list`, if format is `json`, elsewise a `data.frame`

`Analyses.Mutation.MAF` *Retrieve MutSig final analysis MAF.*

Description

This service returns selected columns from the MAF generated by MutSig. These results can optionally be filtered by gene symbol, cohort, tool, and barcode.

Usage

```
Analyses.Mutation.MAF(format = "csv", cohort = "", tool = "MutSig2CV",
  gene = "", tcga_participant_barcode = "", column = "", page = 1,
  page_size = 250, sort_by = "gene")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.
cohort	A character vector indicating the cohort to query, empty string queries all cohorts. See Metadata.Cohorts for available cohorts.
tool	Narrow search to include only data/results produced by the selected Firehose tool. Available tools are provided in the data frame tools.
gene	A character vector of gene symbols. At least one gene OR Barcode is required.
tcga_participant_barcode	A character vector containing TCGA Barcodes. Empty string returns all patients. Either one gene OR barcode is required.
column	Character vector which data columns/fields to return. Every subset of colnames is possible.
page	Subset to be returned.
page_size	Number of records per page, max. is 2000.
sort_by	character indicating the to column which is used for sorting.

Details

MutSig2.0, MutSig2CV (default) and MutSigCV. For more details please visit https://www.broadinstitute.org/cancer/cga/mutsig_faq.

Not specifying a patient-, gene- or cohort subset will return all Variants. This can result in a lot of pages.

Value

A list, if format is json, elsewise a data.frame

Examples

```
format = "json"
cohort = ""
tool = "MutSig2CV"
gene = ""
tcga_participant_barcode = ""
column = ""
page = 1
page_size = 250
sort_by = c("gene")

obj = Analyses.Mutation.MAF(format = format,
                                cohort = cohort,
                                tool = tool,
                                gene =gene,
                                tcga_participant_barcode = tcga_participant_barcode,
                                column = column,
                                page = page,
                                page_size = page_size,
                                sort_by = sort_by)

format = "csv"
obj = Analyses.Mutation.MAF(format = format,
```

```
                                cohort = cohort,
                                tool = tool,
                                gene =gene,
                                tcga_participant_barcode = tcga_participant_barcode,
                                column = column,
                                page = page,
                                page_size = page_size,
                                sort_by = sort_by)

gene = c("TP53", "RUNX1")
tcga_participant_barcode = "TCGA-CH-5761"
cohort = "PRAD"
obj = Analyses.Mutation.MAF(format = format,
                                cohort = cohort,
                                tool = tool,
                                gene =gene,
                                tcga_participant_barcode = tcga_participant_barcode,
                                column = column,
                                page = page,
                                page_size = page_size,
                                sort_by = sort_by)

gene = ""
obj = Analyses.Mutation.MAF(format = format,
                                cohort = cohort,
                                tool = tool,
                                gene = gene,
                                tcga_participant_barcode = tcga_participant_barcode,
                                column = column,
                                page = page,
                                page_size = page_size,
                                sort_by = sort_by)

column = c("tcga_participant_barcode", "Hugo_Symbol", "Variant_Type")
obj = Analyses.Mutation.MAF(format = format,
                                cohort = cohort,
                                tool = tool,
                                gene = gene,
                                tcga_participant_barcode = tcga_participant_barcode,
                                column = column,
                                page = page,
                                page_size = page_size,
                                sort_by = sort_by)
```

`Analyses.Mutation.SMG` *Retrieve Significantly Mutated Genes (SMG).*

Description

This service provides a list of significantly mutated genes. It can be filtered based on date, cohort, rank, gene, tool and/or Q-value threshold.

Usage

```
Analyses.Mutation.SMG(format = "tsv", cohort = "", tool = "MutSig2CV",
  rank = "", gene = "", q = "", page = 1, page_size = 250,
  sort_by = "q")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.
cohort	A character vector indicating the cohort to query, empty string queries all cohorts. See Metadata.Cohorts for available cohorts.
tool	Narrow search to include only data/results produced by the selected Firehose tool. Available tools are provided in the data frame tools.
rank	Number of significant genes to return.
gene	A character vector of gene symbols. At least one gene OR Barcode is required.
q	Only return results with Q-value <= given threshold. For details please see https://www.broadinstitute.org/cancer/cga/mutsig
page	Subset to be returned.
page_size	Number of records per page, max. is 2000.
sort_by	character indicating the to column which is used for sorting.

Value

A list, if format is json, elsewise a data.frame

Examples

```
format = "json"
cohort = "PRAD"
tool = "MutSig2CV"
rank = ""
gene = ""
q = "0.01"
page = 1
page_size = 250
sort_by = "q"


parameters = list(format = format,
                  cohort = cohort,
                  tool = tool,
                  rank = rank,
                  gene = gene,
                  q = q,
                  page = page,
                  page_size = page_size,
                  sort_by = sort_by)

obj = Analyses.Mutation.SMG(format = format,
                                cohort = cohort,
                                tool = tool,
                                rank = rank,
```

```
                                gene = gene,
                                q = q,
                                page = page,
                                page_size = page_size,
                                sort_by = sort_by)

format = "tsv"
obj = Analyses.Mutation.SMG(format = format,
                                cohort = cohort,
                                tool = tool,
                                rank = rank,
                                gene = gene,
                                q = q,
                                page = page,
                                page_size = page_size,
                                sort_by = sort_by)

gene = c("TP53", "SPOP")
q = 0.05
obj = Analyses.Mutation.SMG(format = format,
                                cohort = cohort,
                                tool = tool,
                                rank = rank,
                                gene = gene,
                                q = q,
                                page = page,
                                page_size = page_size,
                                sort_by = sort_by)
gene = ""
obj = Analyses.Mutation.SMG(format = format,
                                cohort = cohort,
                                tool = tool,
                                rank = rank,
                                gene = gene,
                                q = q,
                                page = page,
                                page_size = page_size,
                                sort_by = sort_by)
```

`Analyses.Reports` *Retrieve nozzle report links*

Description

This service returns the nozzle report urls for our Firehose analyses runs. The reports can be filtered based on run date, cohort, report type, and report name.

Usage

```
Analyses.Reports(format = "tsv", date = "", cohort = "", name = "",
  type = "", page = 1, page_size = 250, sort_by = "date")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.
date	Narrow search to one or more Firehose run date stamps. For a list of available dates see Metadata.Dates
cohort	A character vector indicating the cohort to query, empty string queries all cohorts. See Metadata.Cohorts for available cohorts.
name	One or more report names. Please visit http://firebrowse.org/api-docs/#!/Analyses/Reports for details. Currently there is no API call listing all names.
type	One or more report types. Please visit http://firebrowse.org/api-docs/#!/Analyses/Reports for details. Currently there is no API call listing all types.
page	Subset to be returned.
page_size	Number of records per page, max. is 2000.
sort_by	character indicating the to column which is used for sorting.

Value

A list, if format is json, elsewise a data.frame

Archives.StandardData *Retrieve standard data archives.*

Description

This service returns the archive URLs for our Firehose standard data runs, providing a RESTful interface similar in spirit to the command line firehose_get tool. The archives can be filtered based on date, cohort, data type, platform, center, data level, and protocol.

Usage

```
Archives.StandardData(format = "tsv", date = "", cohort = "",
  data_type = "", tool = "", platform = "", center = "", level = "",
  protocol = "", page = 1, page_size = 250, sort_by = "cohort")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.
date	Narrow search to one or more Firehose run date stamps. For a list of available dates see Metadata.Dates
cohort	A character vector indicating the cohort to query, empty string queries all cohorts. See Metadata.Cohorts for available cohorts.
data_type	Narrow search to one or more TCGA data types. Available data types are provided by the data frame data_types.
tool	Narrow search to include only data/results produced by the selected Firehose tool. Available tools are provided in the data frame tools.

`platform`	A platforms name.
`center`	A center name.
`level`	choose data level from 1-4.
`protocol`	Narrow search to one or more sample characterization protocols, see `protocols` data.frame for available protocols.
`page`	Subset to be returned.
`page_size`	Number of records per page, max. is 2000.
`sort_by`	character indicating the to column which is used for sorting.

Details

`data_typem` `tool` and `protocol` are hard coded, since the API provides no methods to query them. Therefor, and to keep always up to date, you may consider looking at the website http://firebrowse.org/api-docs/ for all available data types.

Value

A `list`, if format is `json`, elsewise a `data.frame`

Examples

```
format = "tsv"
date = "2014_12_06"
cohort = "BRCA"
data_type = "CopyNumber"
level = 3
page = 1
page_size = 250

obj = Archives.StandardData(format = format,
                                date = date,
                                cohort = cohort,
                                data_type = data_type,
                                level = level,
                                page = page,
                                page_size = page_size)

cohort = "PRAD"
data_type = "MAF"
level = 3
obj = Archives.StandardData(format = format,
                                date = date,
                                cohort = cohort,
                                data_type = data_type,
                                level = level,
                                page = page,
                                page_size = page_size)
```

Metadata.Centers *Retrieve map of center name(s) to display name(s).*

Description

To return a map of all available centers to display names, exclude the center parameter. Otherwise, choose only the centers you wish to be included in the result.

Usage

```
Metadata.Centers(format = "csv", center = "")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.
center	A center name.

Examples

```
format = "csv"
center = c("intgen.org", "jhu-usc.edu")
obj = Metadata.Centers(format = format, center = center)
```

Metadata.ClinicalTier1 *Retrieve names of all available clinical data elements (CDEs).*

Description

Retrieve names of all available tier 1 clinical data elements (CDEs), unioned across all disease cohorts and patients. For more information on how these CDEs are processed see our pipeline documentation https://confluence.broadinstitute.org/display/GDAC/Documentation

Usage

```
Metadata.ClinicalTier1(format = "csv")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.

Examples

```
format = "csv"
obj = Metadata.ClinicalTier1(format = format)

format = "tsv"
obj = Metadata.ClinicalTier1(format = format)
```

Metadata.Cohort *Retrieve cohort description.*

Description

In this client, the function does, exactly the same as `Metadata.Cohorts`. It just differs within the internal implentation. Maybe, depending on the API, this will change.

Usage

```
Metadata.Cohort(format = "csv", cohort = "")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. `json`, `csv` or `tsv` are available. In this package `tsv` and `csv` are identical.
cohort	A character vector indicating the cohort to query, empty string queries all cohorts. See `Metadata.Cohorts` for available cohorts.

Examples

```
format = "csv"
cohort = c("PRAD", "BRCA")

obj = Metadata.Cohort(format = format,
                      cohort = cohort)

format = "tsv"
cohort = c("PRAD", "BRCA")

obj = Metadata.Cohort(format = format,
                      cohort = cohort)
```

Metadata.Cohorts *Retrieve map of cohort abbreviation(s) to cohort name(s).*

Description

To return a map of all available cohort abbreviations to cohort names, exclude the cohort parameter. Otherwise, choose only the cohort abbreviations you wish to be included in the result.

Usage

```
Metadata.Cohorts(format = "csv", cohort = "")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. `json`, `csv` or `tsv` are available. In this package `tsv` and `csv` are identical.
cohort	A character vector indicating the cohort to query, empty string queries all cohorts. See `Metadata.Cohorts` for available cohorts.

Examples

```
format = "csv"
cohort = c("PRAD", "BRCA")

obj = Metadata.Cohorts(format = format,
                       cohort = cohort)

format = "tsv"
cohort = c("PRAD", "BRCA")

obj = Metadata.Cohorts(format = format,
                       cohort = cohort)
```

Metadata.Counts *Retrieve sample counts.*

Description

Returns the number of available samples, after applying submitted filter.

Usage

```
Metadata.Counts(format = "csv", date = "", cohort = "",
  sample_type = "", data_type = "", totals = T)
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.
date	Narrow search to one or more Firehose run date stamps. For a list of available dates see Metadata.Dates
cohort	A character vector indicating the cohort to query, empty string queries all cohorts. See Metadata.Cohorts for available cohorts.
sample_type	A character vector indicating the sample types to query. Empty string returns all types. See Metadata.SampleTypes, for available sample types.
data_type	Narrow search to one or more TCGA data types. Available data types are provided by the data frame data_types.
totals	Provide an extra column/element giving the total sum of samples. Not implemented by API...

Examples

```
format = "csv"
date = "2015_02_04"
cohort = c("PRAD", "BRCA")
sample_type = ""
data_type = ""
totals = TRUE
obj = Metadata.Counts(format = format,
                              date = date,
                              cohort = cohort,
```

```
                                sample_type = sample_type,
                                data_type = data_type,
                                totals = totals)

format = "tsv"
cohort = c("PRAD", "BRCA")
sample_type = "Tumor"
data_type = c("methylation", "maf")
totals = TRUE

obj = Metadata.Counts(format = format,
                                date = date,
                                cohort = cohort,
                                sample_type = sample_type,
                                data_type = data_type,
                                totals = totals)
```

`Metadata.Dates` *Retrieve list of Firehose standard data and analyses dates.*

Description

Retrieve list of Firehose standard data and analyses dates.

Usage

```
Metadata.Dates(format = "csv")
```

Arguments

`format`	A string identifying the data type returned. Using json requires the jsonlite package. `json`, `csv` or `tsv` are available. In this package `tsv` and `csv` are identical.

Examples

```
format = "csv"
obj = Metadata.Dates(format = format)

format = "tsv"
obj = Metadata.Dates(format = format)
```

`Metadata.HeartBeat` *Simple way to discern whether API server is up and running*

Description

Returns a message to indicate that API (server) is up and running, or times out if not.

Usage

```
Metadata.HeartBeat(format = "csv")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.

Examples

```
format = "csv"
obj = Metadata.HeartBeat(format = format)
```

Metadata.Platforms *Retrieve map of platform code(s) to platform name(s).*

Description

To return a map of all available platform codes to platform names, exclude the platform parameter. Otherwise, choose only the platforms you wish to be included in the result.

Usage

```
Metadata.Platforms(format = "csv", platform = "")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.
platform	A platforms name.

Examples

```
format = "csv"
obj = Metadata.Platforms(format = format)

format = "tsv"
obj = Metadata.Platforms(format = format)
platform = c("454", "ABI", "biotab")
obj = Metadata.Platforms(format = format,
                         platform = platform)
```

Metadata.SampleType.Barcode

Return a sample type code to short letter code mapping.

Description

Return a sample type code to short letter code mapping.

Usage

```
Metadata.SampleType.Barcode(format = "csv", tcga_participant_barcode = "")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.
tcga_participant_barcode	A character vector containing TCGA Barcodes. Empty string returns all patients. Either one gene OR barcode is required.

Examples

```
format = "csv"
tcga_participant_barcode = c("TCGA-GF-A4EO-06",
                             "TCGA-EL-A3D5-01A-22D-A202-08")
obj = Metadata.SampleType.Barcode(format = format,
                                  tcga_participant_barcode = tcga_participant_barcode)
```

Metadata.SampleType.Code

Translate from numeric sample type code to short letter code.

Description

Translate a numeric sample type code (e.g. 01, 02, etc.) into its short letter code (e.g. TP, TR, etc.).

Usage

```
Metadata.SampleType.Code(format = "csv", code = "")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.
code	Sample type codes.

Examples

```
format = "csv"
code = c(50, 60)
obj = Metadata.SampleType.Code(format = format, code = code)
```

Metadata.SampleType.ShortLetterCode

Translate from sample type short letter code to numeric code.

Description

Translate a sample type short letter code (e.g. TP, TR, etc.) into its numeric sample type code (e.g. 01, 02, etc.).

Usage

```
Metadata.SampleType.ShortLetterCode(format = "csv", short_letter_code = "")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.
short_letter_code	Short Letter Code e.g. TP, TM, NB, ...

Examples

```
format = "csv"
short_letter_code = c("TR", "TR")
obj = Metadata.SampleType.ShortLetterCode(format = format,
short_letter_code = short_letter_code)
```

Metadata.SampleTypes *Return a sample type code to short letter code mapping.*

Description

Return a sample type code to short letter code mapping.

Usage

```
Metadata.SampleTypes(format = "csv")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.

Examples

```
format = "csv"
obj = Metadata.SampleTypes(format = format)

format = "tsv"
obj = Metadata.SampleTypes(format = format)
```

Samples.ClinicalTier1 *Retrieve clinical pick tier 1 data.*

Description

This service returns patient-level tier 1 clinical data elements (CDEs). Results may be filtered by disease cohort, patient barcode or CDE name. When filtering by CDE note that only when a patient record contains one or more of the selected CDEs will it be returned. For more information on how these CDEs are processed see our pipeline documentation https://confluence.broadinstitute.org/display/GDAC/Documentation#Documentation-ClinicalPipeline.

Usage

```
Samples.ClinicalTier1(format = "csv", cohort = "",
  tcga_participant_barcode = "", tier1_cde_name = "", page = 1,
  page_size = 250, sort_by = "cohort")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.
cohort	A character vector indicating the cohort to query, empty string queries all cohorts. See Metadata.Cohorts for available cohorts.
tcga_participant_barcode	A character vector containing TCGA Barcodes. Empty string returns all patients. Either one gene OR barcode is required.
tier1_cde_name	A character vector containing the CDEs which should be used for subsetting. To view all CDEs see Metadata.ClinicalTier1.
page	Subset to be returned.
page_size	Number of records per page, max. is 2000.
sort_by	character indicating the to column which is used for sorting.

Value

A list, if format is json, elsewise a data.frame

Examples

```
format = "json"
cohort = ""
tcga_participant_barcode = ""
tier1_cde_name = ""
page = 1
page_size = 250
sort_by = "cohort"

obj = Samples.ClinicalTier1(format = format,
                            cohort = cohort,
                            tcga_participant_barcode = tcga_participant_barcode,
                            tier1_cde_name = tier1_cde_name,
                            page = page,
```

```
                                page_size = page_size,
                                sort_by = sort_by)

format = "csv"
obj = Samples.ClinicalTier1(format = format,
                                cohort = cohort,
                                tcga_participant_barcode = tcga_participant_barcode,
                                tier1_cde_name = tier1_cde_name,
                                page = page,
                                page_size = page_size,
                                sort_by = sort_by)

tcga_participant_barcode = "TCGA-GF-A4EO"
obj = Samples.ClinicalTier1(format = "tsv",
                                cohort = cohort,
                                tcga_participant_barcode = tcga_participant_barcode,
                                tier1_cde_name = tier1_cde_name,
                                page = page,
                                page_size = page_size,
                                sort_by = sort_by)
```

`Samples.miRSeq` *Retrieve miRSeq data*

Description

This service returns sample-level log2 miRSeq expression values. Results may be filtered by miR, cohort, barcode, sample type or Firehose preprocessing tool, but at least one miR OR barcode must be supplied.

Usage

```
Samples.miRSeq(format = "csv", mir = "", cohort = "",
  tcga_participant_barcode = "", tool = "miRseq_Mature_Preprocess",
  sample_type = "", page = 1, page_size = 250, sort_by = "mir")
```

Arguments

`format`	A string identifying the data type returned. Using json requires the jsonlite package. `json`, `csv` or `tsv` are available. In this package `tsv` and `csv` are identical.
`mir`	A character vector of miR names. At least one miR or barcode is required.
`cohort`	A character vector indicating the cohort to query, empty string queries all cohorts. See `Metadata.Cohorts` for available cohorts.
`tcga_participant_barcode`	A character vector containing TCGA Barcodes. Empty string returns all patients. Either one gene OR barcode is required.
`tool`	The tool used to run the anylses, either `miRseq_Mature_Preprocess` or `miRseq_Preprocess`
`sample_type`	A character vector indicating the sample types to query. Empty string returns all types. See `Metadata.SampleTypes`, for available sample types.
`page`	Subset to be returned.
`page_size`	Number of records per page, max. is 2000.
`sort_by`	character indicating the to column which is used for sorting.

Value

A `list`, if format is `json`, elsewise a `data.frame`

Examples

```
format = "json"
mir = c("hsa-mir-1285-3p","hsa-mir-125a-5p","hsa-mir-221-3p",
        "hsa-mir-10b-5p","hsa-mir-608","hsa-mir-324-5p")
cohort = "BRCA"
tcga_participant_barcode = ""
tool = "miRseq_Mature_Preprocess"
sample_type = "NT"
page = 1
page_size = 250
sort_by = "mir"

# Get results in json format/list
obj = Samples.miRSeq(format = format,
                        mir = mir,
                        cohort = cohort,
                        tcga_participant_barcode = tcga_participant_barcode,
                        tool = tool,
                        sample_type = sample_type,
                        page = page,
                        page_size = page_size,
                        sort_by = sort_by)

# Nor as CSV
format = "csv"
mir = c("hsa-mir-1285-3p", "hsa-mir-125a-5p")
obj = Samples.miRSeq(format = format,
                        mir = mir,
                        cohort = cohort,
                        tcga_participant_barcode = tcga_participant_barcode,
                        tool = tool,
                        sample_type = sample_type,
                        page = page,
                        page_size = page_size,
                        sort_by = sort_by)

# And just by miR IDs, without any other restrictions
obj = Samples.miRSeq(mir = c("hsa-mir-1285-3p","hsa-mir-125a-5p"))
```

`Samples.mRNASeq` *Retrieve mRNASeq data*

Description

This service returns sample-level log2 mRNASeq expression values. Results may be filtered by gene, cohort, barcode, sample type or characterization protocol, but at least one gene OR barcode must be supplied.

Usage

```
Samples.mRNASeq(format = "csv", gene = "", cohort = "",
  tcga_participant_barcode = "", sample_type = "", protocol = "RSEM",
  page = 1, page_size = 250, sort_by = "gene")
```

Arguments

format	A string identifying the data type returned. Using json requires the jsonlite package. json, csv or tsv are available. In this package tsv and csv are identical.
gene	A character vector of gene symbols. At least one gene OR Barcode is required.
cohort	A character vector indicating the cohort to query, empty string queries all cohorts. See Metadata.Cohorts for available cohorts.
tcga_participant_barcode	A character vector containing TCGA Barcodes. Empty string returns all patients. Either one gene OR barcode is required.
sample_type	A character vector indicating the sample types to query. Empty string returns all types. See Metadata.SampleTypes, for available sample types.
protocol	Should RSEM (default) or RPKM data be retrieved.
page	Subset to be returned.
page_size	Number of records per page, max. is 2000.
sort_by	character indicating the to column which is used for sorting.

Details

For furhter information please see https://confluence.broadinstitute.org/display/GDAC/FAQ

Value

A list, if format is json, elsewise a data.frame

Examples

```
format = "json"
gene = c("AKT3", "RB1", "MAP3K1")
cohort = "BRCA"
tcga_participant_barcode = ""
sample_type = "NT"
protocol = "RSEM"
page = 1
page_size = 250
sort_by = "gene"
obj = Samples.mRNASeq(format = format, gene, cohort, tcga_participant_barcode,
                          sample_type, protocol, page, page_size, sort_by)
# returns a list from json

obj = Samples.mRNASeq(gene = c("TP53", "RUNX1"))
# returns a data frame, default
```

Index