

# Package ‘cBioportalR’

October 12, 2022

**Title** Browse and Query Clinical and Genomic Data from cBioPortal

**Version** 1.0.1

**Description** Provides R users with direct access to genomic and clinical data from the 'cBioPortal' web resource via user-friendly functions that wrap 'cBioPortal's' existing API endpoints <<https://www.cbioportal.org/api/swagger-ui/index.html>>. Users can browse and query genomic data on mutations, copy number alterations and fusions, as well as data on tumor mutational burden ('TMB'), microsatellite instability status ('MSI'), 'FACETS' and select clinical data points (depending on the study).

See <<https://www.cbioportal.org/>> and Gao et al., (2013) <[doi:10.1126/scisignal.2004088](https://doi.org/10.1126/scisignal.2004088)> for more information on the cBioPortal web resource.

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**Suggests** testthat (>= 3.1.4), knitr (>= 1.39), rmarkdown (>= 2.14), covr (>= 3.5.1), spelling (>= 2.2)

**Depends** R (>= 2.10)

**Imports** httr (>= 1.4.3), tibble (>= 3.1.7), purrr (>= 0.3.4), magrittr (>= 2.0.3), rlang (>= 1.0.3), glue (>= 1.6.2), jsonlite (>= 1.8.0), tidyr (>= 1.2.0), dplyr (>= 1.0.9), stringr (>= 1.4.0), cli (>= 3.3.0)

**RoxygenNote** 7.2.1

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**URL** <https://github.com/karissawhiting/cbioportalR>,  
<https://www.karissawhiting.com/cbioportalR/>

**BugReports** <https://github.com/karissawhiting/cbioportalR/issues>

**Language** en-US

**NeedsCompilation** no

**Author** Karissa Whiting [aut, cre] (<<https://orcid.org/0000-0002-4683-1868>>)

**Maintainer** Karissa Whiting <karissa.whiting@gmail.com>

**Repository** CRAN

**Date/Publication** 2022-08-29 06:20:02 UTC

## R topics documented:

available_clinical_attributes . . . . .	3
available_gene_panels . . . . .	3
available_patients . . . . .	4
available_profiles . . . . .	5
available_samples . . . . .	5
available_studies . . . . .	6
get_alias . . . . .	7
get_cbioportal_token . . . . .	7
get_clinical_by_patient . . . . .	8
get_clinical_by_sample . . . . .	9
get_clinical_by_study . . . . .	10
get_cna_by_sample . . . . .	11
get_cna_by_study . . . . .	13
get_entrez_id . . . . .	14
get_fusions_by_sample . . . . .	14
get_fusions_by_study . . . . .	16
get_genes . . . . .	17
get_genetics_by_sample . . . . .	17
get_genetics_by_study . . . . .	19
get_gene_panel . . . . .	19
get_hugo_symbol . . . . .	20
get_mutations_by_sample . . . . .	21
get_mutations_by_study . . . . .	22
get_panel_by_sample . . . . .	23
get_samples_by_patient . . . . .	24
get_study_info . . . . .	25
impact_gene_info . . . . .	25
set_cbioportal_db . . . . .	26
test_cbioportal_db . . . . .	27
<b>Index</b>	<b>28</b>

---

```
available_clinical_attributes
```

*Get all available clinical attribute IDs for a study*

---

**Description**

Get all available clinical attribute IDs for a study

**Usage**

```
available_clinical_attributes(study_id = NULL, base_url = NULL)
```

**Arguments**

study_id	cbioportal study ID
base_url	The database URL to query. If NULL will default to URL set with set_cbioportal_db(<your_db>)

**Value**

a data frame of available clinical attributes for that study

**Examples**

```
## Not run:  
available_clinical_attributes("acc_tcga", base_url = 'www.cbioportal.org/api')  
  
## End(Not run)
```

---

```
available_gene_panels
```

*Get Available Gene Panels For a Database*

---

**Description**

Get Available Gene Panels For a Database

**Usage**

```
available_gene_panels(base_url = NULL)
```

**Arguments**

base_url	The database URL to query. If NULL will default to URL set with set_cbioportal_db(<your_db>)
----------	--

**Value**

a dataframe of metadata regarding each available panel

**Examples**

```
## Not run:  
set_cbioportal_db("public")  
available_gene_panels()  
  
## End(Not run)
```

---

available_patients	<i>Get All Patient IDs in a Study</i>
--------------------	---------------------------------------

---

**Description**

Get All Patient IDs in a Study

**Usage**

```
available_patients(study_id = NULL, base_url = NULL)
```

**Arguments**

study_id	A character string indicating which study ID should be searched. Only 1 study allowed. If NULL, we will guess a default study ID based on your database URL.
base_url	The database URL to query If NULL will default to URL set with set_cbioportal_db(<your_db>)

**Value**

A dataframe of patient\_ids in a given study

**Examples**

```
## Not run:  
set_cbioportal_db("public")  
available_samples(study_id = "acc_tcga")  
  
## End(Not run)
```

---

available_profiles	<i>Get Available Genomic Profiles For a Study or Database</i>
--------------------	---

---

**Description**

Get Available Genomic Profiles For a Study or Database

**Usage**

```
available_profiles(study_id = NULL, base_url = NULL)
```

**Arguments**

study_id	A character vector of length 1 indicating study_id. See <code>get_studies()</code> to see all available studies for your URL. If NULL, it will return all profiles for your current database url
base_url	The database URL to query. If NULL will default to URL set with <code>set_cbioportal_db(&lt;your_db&gt;)</code>

**Value**

A dataframe of available genetic profiles and their profile ids

**Examples**

```
## Not run:  
set_cbioportal_db("public")  
available_profiles()  
available_profiles(study_id = "acc_tcga")  
  
## End(Not run)
```

---

available_samples	<i>Get All Sample IDs in a Study</i>
-------------------	--------------------------------------

---

**Description**

Get All Sample IDs in a Study

**Usage**

```
available_samples(study_id = NULL, base_url = NULL)
```

**Arguments**

study_id	A character string indicating which study ID should be searched. Only 1 study allowed. If NULL, we will guess a default study ID based on your database URL.
base_url	The database URL to query If NULL will default to URL set with <code>set_cbioportal_db(&lt;your_db&gt;)</code>

**Value**

A dataframe of sample\_ids in a given study

**Examples**

```
## Not run:  
set_cbioportal_db("public")  
available_samples(study_id = "acc_tcga")  
  
## End(Not run)
```

---

available_studies	<i>Get Metadata on All Available Studies in a Database</i>
-------------------	--

---

**Description**

Get Metadata on All Available Studies in a Database

**Usage**

```
available_studies(base_url = NULL)
```

**Arguments**

base\_url            The database URL to query. If NULL will default to URL set with set\_cbioportal\_db(<your\_db>)

**Value**

A dataframe of available studies and their metadata

**Examples**

```
## Not run:  
set_cbioportal_db("public")  
available_studies()  
  
## End(Not run)
```

---

get_alias	<i>Get Gene Name Alias for a Given Hugo Symbol</i>
-----------	--

---

### Description

This function grabs known gene aliases for a given Hugo Symbol. You may notice that genes - alias pairs are not always consistent. For example `get_alias("KMT2D")` will return "MLL2" but `get_alias("MLL2")` will not return "KMT2D" This function relies on the existing cBioPortal API which controls this database of aliases. Therefore, this is a convenience function but you may want to consider a more carefully curated alias list like `cbioportalR::impact_gene_info`

### Usage

```
get_alias(hugo_symbol = NULL, base_url = NULL)
```

### Arguments

hugo_symbol	a hugo symbol for which to return aliases
base_url	The database URL to query

### Value

A character string with all aliases

### Examples

```
## Not run:  
  
get_alias(hugo_symbol = "FGFR3", base_url = 'www.cbioportal.org/api')  
get_alias(hugo_symbol = c("FGFR3", "TP53"), base_url = 'www.cbioportal.org/api')  
  
## End(Not run)
```

---

get_cbioportal_token	<i>Get cBioPortal Access Token</i>
----------------------	------------------------------------

---

### Description

Convenience function that retrieves cBioPortal token System Environment variable "CBIOPORTAL\_TOKEN"

### Usage

```
get_cbioportal_token()
```

**Value**

Returns a string with cBioPortal token if successfully authenticated, or a warning that token could not be found.

**Author(s)**

Karissa Whiting, Daniel D. Sjoberg

**Examples**

```
## Not run:  
get_cbioportal_token()  
  
## End(Not run)
```

---

get\_clinical\_by\_patient

*Get clinical data by attribute, study ID and patient ID*

---

**Description**

Get clinical data by attribute, study ID and patient ID

**Usage**

```
get_clinical_by_patient(  
  study_id = NULL,  
  patient_id = NULL,  
  patient_study_pairs = NULL,  
  clinical_attribute = NULL,  
  base_url = NULL  
)
```

**Arguments**

study_id	A string indicating the study ID from which to pull data. If no study ID, will guess the study ID based on your URL and inform. Only 1 study ID can be passed. If mutations/cna from more than 1 study needed, see <code>sample_study_pairs</code>
patient_id	a cBioPortal <code>patient_id</code>
patient_study_pairs	A dataframe with columns: <code>patient_id</code> , <code>study_id</code> . Variations in capitalization of column names are accepted. This can be used in place of <code>patient_id</code> , <code>study_id</code> , arguments above if you need to pull samples from several different studies at once. If passed, this will take over <code>patient_id</code> and <code>study_id</code> if they are also passed.



`clinical_attribute` one or more clinical attributes for your study. If none provided, will return all attributes available for studies

`base_url` The database URL to query If NULL will default to URL set with `set_cbioportal_db(<your_db>)`

**Value**

a dataframe of a specific clinical attribute

**Examples**

```
## Not run:

ex <- tibble::tribble(
  ~patientID, ~study_id,
  "P-0001453", "blca_nmIBC_2017",
  "P-0002166", "blca_nmIBC_2017",
  "P-0003238", "blca_nmIBC_2017",
  "P-0000004", "msk_impact_2017",
  "P-0000023", "msk_impact_2017")

x <- get_clinical_by_patient(patient_study_pairs = ex,
  clinical_attribute = NULL, base_url = 'www.cbioportal.org/api')

## End(Not run)
```

---

`get_clinical_by_sample`

*Get clinical data by attribute, study ID and sample ID*

---

**Description**

Get clinical data by attribute, study ID and sample ID

**Usage**

```
get_clinical_by_sample(
  study_id = NULL,
  sample_id = NULL,
  sample_study_pairs = NULL,
  clinical_attribute = NULL,
  base_url = NULL
)
```

**Arguments**

study_id	A string indicating the study ID from which to pull data. If no study ID, will guess the study ID based on your URL and inform. Only 1 study ID can be passed. If mutations/cna from more than 1 study needed, see sample_study_pairs
sample_id	a vector of sample IDs (character)
sample_study_pairs	A dataframe with columns: sample_id, study_id and molecular_profile_id (optional). Variations in capitalization of column names are accepted. This can be used in place of sample_id, study_id, molecular_profile_id arguments above if you need to pull samples from several different studies at once. If passed this will take overwrite sample_id, study_id, molecular_profile_id if also passed.
clinical_attribute	one or more clinical attributes for your study. If none provided, will return all attributes available for studies
base_url	The database URL to query If NULL will default to URL set with set_cbioportal_db(<your_db>)

**Value**

a dataframe of a specific clinical attribute

**Examples**

```
## Not run:
get_clinical_by_sample(study_id = "acc_tcga", sample_id = "TCGA-OR-A5J2-01",
  clinical_attribute = "CANCER_TYPE", base_url = 'www.cbioportal.org/api')

ex <- tibble::tribble(
  ~sample_id, ~study_id,
  "P-0001453-T01-IM3", "blca_nmIBC_2017",
  "P-0002166-T01-IM3", "blca_nmIBC_2017",
  "P-0003238-T01-IM5", "blca_nmIBC_2017",
  "P-0000004-T01-IM3", "msk_impact_2017",
  "P-0000023-T01-IM3", "msk_impact_2017")

x <- get_clinical_by_sample(sample_study_pairs = ex,
  clinical_attribute = NULL, base_url = 'www.cbioportal.org/api')

## End(Not run)
```

---

get\_clinical\_by\_study *Get all available clinical data for a specified study*

---

**Description**

Returns all sample-level and patient-level clinical data for a given study

**Usage**

```
get_clinical_by_study(
  study_id = NULL,
  clinical_attribute = NULL,
  base_url = NULL
)
```

**Arguments**

```
study_id      study ID
clinical_attribute
               one or more clinical attributes for your study. If none provided, will return all at-
               tributes available for that study (available_clinical_attributes(<study_id>))
base_url      The database URL to query. If NULL will default to URL set with set_cbioportal_db(<your_db>)
```

**Value**

a dataframe of all available clinical attributes and their values

**Examples**

```
## Not run:
get_clinical_by_study(study_id = "acc_tcga",
  clinical_attribute = "CANCER_TYPE", base_url = 'www.cbioportal.org/api')

get_clinical_by_study(study_id = "acc_tcga", base_url = 'www.cbioportal.org/api')

## End(Not run)
```

---

```
get_cna_by_sample      Get CNA By Sample ID
```

---

**Description**

Get CNA By Sample ID

**Usage**

```
get_cna_by_sample(
  sample_id = NULL,
  study_id = NULL,
  molecular_profile_id = NULL,
  sample_study_pairs = NULL,
  genes = NULL,
  panel = NULL,
  add_hugo = TRUE,
  base_url = NULL
)
```

**Arguments**

sample_id	a vector of sample IDs (character)
study_id	A string indicating the study ID from which to pull data. If no study ID, will guess the study ID based on your URL and inform. Only 1 study ID can be passed. If mutations/cna from more than 1 study needed, see sample_study_pairs
molecular_profile_id	A string indicating the molecular profile ID from which to pull data. If ID supplied, will guess the molecular profile ID based on the study ID. Only 1 molecular profile ID can be passed. If mutations from more than 1 study needed, see sample_study_pairs
sample_study_pairs	A dataframe with columns: sample_id, study_id and molecular_profile_id (optional). Variations in capitalization of column names are accepted. This can be used in place of sample_id, study_id, molecular_profile_id arguments above if you need to pull samples from several different studies at once. If passed this will take overwrite sample_id, study_id, molecular_profile_id if also passed.
genes	A vector of Entrez ids or Hugo symbols. If Hugo symbols are supplied, they will be converted to entrez ids using the get_entrez_id() function. If panel and genes are both supplied, genes from both arguments will be returned. If both are NULL (default), it will return gene results for all available genomic data for that sample.
panel	One or more panel IDs to query (e.g. 'IMPACT468'). If panel and genes are both supplied, genes from both arguments will be returned. If both are NULL (default), it will return gene results for all available genomic data for that sample.
add_hugo	Logical indicating whether HugoGeneSymbol should be added to your resulting data frame, if not already present in raw API results. Argument is TRUE by default. If FALSE, results will be returned as is (i.e. any existing Hugo Symbol columns in raw results will not be removed).
base_url	The database URL to query If NULL will default to URL set with set_cbioportal_db(<your_db>)

**Value**

A data frame of CNAs

**Examples**

```
## Not run:
set_cbioportal_db("public")
get_cna_by_sample(sample_id = c("s_C_36924L_P001_d"),
                  study_id = "prad_msk_2019")

## End(Not run)
```

---

get_cna_by_study	<i>Get CNA By Study</i>
------------------	-------------------------

---

## Description

Get CNA By Study

## Usage

```
get_cna_by_study(  
  study_id = NULL,  
  molecular_profile_id = NULL,  
  add_hugo = TRUE,  
  base_url = NULL  
)
```

## Arguments

study_id	A study ID to query mutations. If NULL, guesses study ID based on molecular_profile_id.
molecular_profile_id	a molecular profile to query mutations. If NULL, guesses molecular_profile_id based on study ID.
add_hugo	Logical indicating whether HugoGeneSymbol should be added to your resulting data frame, if not already present in raw API results. Argument is TRUE by default. If FALSE, results will be returned as is (i.e. any existing Hugo Symbol columns in raw results will not be removed).
base_url	The database URL to query If NULL will default to URL set with set_cbiportal_db(<your_db>)

## Value

A dataframe of CNAs

## Examples

```
## Not run:  
get_cna_by_study(study_id = "prad_msk_2019")  
get_cna_by_study(molecular_profile_id = "prad_msk_2019_cna")  
  
## End(Not run)
```

---

get\_entrez\_id                    *Get Entrez Gene ID for a given set of Hugo Symbols*

---

**Description**

Get Entrez Gene ID for a given set of Hugo Symbols

**Usage**

```
get_entrez_id(hugo_symbol = NULL, base_url = NULL)
```

**Arguments**

hugo\_symbol        a character vector of Hugo Symbols  
base\_url            The database URL to query

**Value**

A dataframe with Entrez Gene IDs and Hugo Symbols

**Examples**

```
## Not run:  
get_entrez_id(hugo_symbol = "TAP1", base_url = 'www.cbioportal.org/api')  
get_entrez_id(hugo_symbol = c("FGFR1", "TP53"), base_url = 'www.cbioportal.org/api')  
  
## End(Not run)
```

---

get\_fusions\_by\_sample    *Get Fusions By Sample ID*

---

**Description**

Get Fusions By Sample ID

**Usage**

```
get_fusions_by_sample(  
  sample_id = NULL,  
  study_id = NULL,  
  molecular_profile_id = NULL,  
  sample_study_pairs = NULL,  
  genes = NULL,  
  panel = NULL,  
  base_url = NULL  
)
```

```

get_structural_variants_by_sample(
  sample_id = NULL,
  study_id = NULL,
  molecular_profile_id = NULL,
  sample_study_pairs = NULL,
  genes = NULL,
  panel = NULL,
  base_url = NULL
)

```

### Arguments

sample_id	a vector of sample IDs (character)
study_id	A string indicating the study ID from which to pull data. If no study ID, will guess the study ID based on your URL and inform. Only 1 study ID can be passed. If mutations/cna from more than 1 study needed, see sample_study_pairs
molecular_profile_id	A string indicating the molecular profile ID from which to pull data. If ID supplied, will guess the molecular profile ID based on the study ID. Only 1 molecular profile ID can be passed. If mutations from more than 1 study needed, see sample_study_pairs
sample_study_pairs	A dataframe with columns: sample_id, study_id and molecular_profile_id (optional). Variations in capitalization of column names are accepted. This can be used in place of sample_id, study_id, molecular_profile_id arguments above if you need to pull samples from several different studies at once. If passed this will take overwrite sample_id, study_id, molecular_profile_id if also passed.
genes	A vector of Entrez ids or Hugo symbols. If Hugo symbols are supplied, they will be converted to entrez ids using the get_entrez_id() function. If panel and genes are both supplied, genes from both arguments will be returned. If both are NULL (default), it will return gene results for all available genomic data for that sample.
panel	One or more panel IDs to query (e.g. 'IMPACT468'). If panel and genes are both supplied, genes from both arguments will be returned. If both are NULL (default), it will return gene results for all available genomic data for that sample.
base_url	The database URL to query If NULL will default to URL set with set_cbioportal_db(<your_db>)

### Value

A data frame of Fusions

### Examples

```

## Not run:
set_cbioportal_db("public")

```

```

#' # These return the same results
get_fusions_by_sample(sample_id = c("s_C_CAUWT7_P001_d"),
                      study_id = "prad_msk_2019")
get_structural_variants_by_sample(sample_id = c("s_C_CAUWT7_P001_d"),
                                  study_id = "prad_msk_2019")

## End(Not run)

```

---

get\_fusions\_by\_study *Get Fusions By Study*

---

## Description

Get Fusions By Study

## Usage

```

get_fusions_by_study(
  study_id = NULL,
  molecular_profile_id = NULL,
  base_url = NULL
)

get_structural_variants_by_study(
  study_id = NULL,
  molecular_profile_id = NULL,
  base_url = NULL
)

```

## Arguments

study_id	A study ID to query mutations. If NULL, guesses study ID based on molecular_profile_id.
molecular_profile_id	a molecular profile to query mutations. If NULL, guesses molecular_profile_id based on study ID.
base_url	The database URL to query If NULL will default to URL set with set_cbiportal_db(<your_db>)

## Value

A dataframe of fusions



**Examples**

```
## Not run:  
# These return the same results  
get_fusions_by_study(molecular_profile_id = "prad_msk_2019_structural_variants")  
  
get_structural_variants_by_study(molecular_profile_id =  
  "prad_msk_2019_structural_variants")  
  
## End(Not run)
```

---

`get_genes`*Get A List of Genes for a Specified Database*

---

**Description**

Get A List of Genes for a Specified Database

**Usage**

```
get_genes(base_url = NULL)
```

**Arguments**

`base_url` The database URL to query If NULL will default to URL set with `set_cbioportal_db(<your_db>)`

**Value**

A dataframe of gene ids, hugo symbols, and gene types

**Examples**

```
## Not run:  
get_genes(base_url = 'www.cbioportal.org/api')  
  
## End(Not run)
```

---

`get_genetics_by_sample`*Get All Genomic Information By Sample IDs*

---

**Description**

Get All Genomic Information By Sample IDs

**Usage**

```

get_genetics_by_sample(
  sample_id = NULL,
  study_id = NULL,
  sample_study_pairs = NULL,
  genes = NULL,
  panel = NULL,
  add_hugo = TRUE,
  base_url = NULL
)

```

**Arguments**

<code>sample_id</code>	a vector of sample IDs (character)
<code>study_id</code>	A string indicating the study ID from which to pull data. If no study ID, will guess the study ID based on your URL and inform. Only 1 study ID can be passed. If mutations/cna from more than 1 study needed, see <code>sample_study_pairs</code>
<code>sample_study_pairs</code>	A dataframe with columns: <code>sample_id</code> , <code>study_id</code> and <code>molecular_profile_id</code> (optional). Variations in capitalization of column names are accepted. This can be used in place of <code>sample_id</code> , <code>study_id</code> , <code>molecular_profile_id</code> arguments above if you need to pull samples from several different studies at once. If passed this will take over <code>sample_id</code> , <code>study_id</code> , <code>molecular_profile_id</code> if also passed.
<code>genes</code>	A vector of Entrez ids or Hugo symbols. If Hugo symbols are supplied, they will be converted to entrez ids using the <code>get_entrez_id()</code> function. If <code>panel</code> and <code>genes</code> are both supplied, genes from both arguments will be returned. If both are NULL (default), it will return gene results for all available genomic data for that sample.
<code>panel</code>	One or more panel IDs to query (e.g. 'IMPACT468'). If <code>panel</code> and <code>genes</code> are both supplied, genes from both arguments will be returned. If both are NULL (default), it will return gene results for all available genomic data for that sample.
<code>add_hugo</code>	Logical indicating whether HugoGeneSymbol should be added to your resulting data frame, if not already present in raw API results. Argument is TRUE by default. If FALSE, results will be returned as is (i.e. any existing Hugo Symbol columns in raw results will not be removed).
<code>base_url</code>	The database URL to query If NULL will default to URL set with <code>set_cbiportal_db(&lt;your_db&gt;)</code>

**Value**

A list of mutations, cna and structural variants (including fusions), if available.

**Examples**

```

## Not run:
get_genetics_by_sample(sample_id = c("TCGA-OR-A5J2-01", "TCGA-OR-A5J6-01"),
  study_id = "acc_tcga")

```

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

---

get\_genetics\_by\_study *Get All Genomic Information By Study*

---

### Description

Get All Genomic Information By Study

### Usage

```
get_genetics_by_study(study_id = NULL, add_hugo = TRUE, base_url = NULL)
```

### Arguments

study_id	A study ID to query mutations. If NULL, guesses study ID based on molecular_profile_id.
add_hugo	Logical indicating whether HugoGeneSymbol should be added to your resulting data frame, if not already present in raw API results. Argument is TRUE by default. If FALSE, results will be returned as is (i.e. any existing Hugo Symbol columns in raw results will not be removed).
base_url	The database URL to query If NULL will default to URL set with set_cbioportal_db(<your_db>)

### Value

A list of mutations, cna and structural variants (including fusions), if available.

### Examples

```
## Not run:  
get_genetics_by_study(study_id = "prad_msk_2019")  
  
## End(Not run)
```

---

get\_gene\_panel *Retrieve Genes Included For a Specified Panel ID*

---

### Description

Retrieve Genes Included For a Specified Panel ID

### Usage

```
get_gene_panel(panel_id = NULL, base_url = NULL)
```

**Arguments**

panel\_id            name of panel. See available\_gene\_panels() to get panel ID  
 base\_url            The database URL to query. If NULL will default to URL set with set\_cbioportal\_db(<your\_db>)

**Value**

A dataframe of genes in a specified panel

**Examples**

```
## Not run:
get_gene_panel(panel_id = "IMPACT468", base_url = 'www.cbioportal.org/api')

## End(Not run)
```

---

get\_hugo\_symbol            *Get Hugo Symbol for a given set of Entrez IDs*

---

**Description**

Get Hugo Symbol for a given set of Entrez IDs

**Usage**

```
get_hugo_symbol(entrez_id = NULL, base_url = NULL)
```

**Arguments**

entrez\_id            a character or numeric vector of Entrez gene IDs  
 base\_url            The database URL to query

**Value**

A dataframe with Entrez Gene IDs and Hugo Symbols

**Examples**

```
## Not run:
get_hugo_symbol(entrez_id = 2261, base_url = 'www.cbioportal.org/api')
get_hugo_symbol(entrez_id = c(2261, 7157) , base_url = 'www.cbioportal.org/api')

## End(Not run)
```

---

`get_mutations_by_sample`*Get Mutations By Sample ID*

---

## Description

Get Mutations By Sample ID

## Usage

```
get_mutations_by_sample(  
  sample_id = NULL,  
  study_id = NULL,  
  molecular_profile_id = NULL,  
  sample_study_pairs = NULL,  
  genes = NULL,  
  panel = NULL,  
  add_hugo = TRUE,  
  base_url = NULL  
)
```

## Arguments

- |                                   |  |
|-----------------------------------|--|
| <code>sample_id</code>            | a vector of sample IDs (character)   |
| <code>study_id</code>             | A string indicating the study ID from which to pull data. If no study ID, will guess the study ID based on your URL and inform. Only 1 study ID can be passed. If mutations/cna from more than 1 study needed, see <code>sample_study_pairs</code>   |
| <code>molecular_profile_id</code> | A string indicating the molecular profile ID from which to pull data. If ID supplied, will guess the molecular profile ID based on the study ID. Only 1 molecular profile ID can be passed. If mutations from more than 1 study needed, see <code>sample_study_pairs</code>  |
| <code>sample_study_pairs</code>   | A dataframe with columns: <code>sample_id</code> , <code>study_id</code> and <code>molecular_profile_id</code> (optional). Variations in capitalization of column names are accepted. This can be used in place of <code>sample_id</code> , <code>study_id</code> , <code>molecular_profile_id</code> arguments above if you need to pull samples from several different studies at once. If passed this will take over <code>sample_id</code> , <code>study_id</code> , <code>molecular_profile_id</code> if also passed. |
| <code>genes</code>                | A vector of Entrez ids or Hugo symbols. If Hugo symbols are supplied, they will be converted to entrez ids using the <code>get_entrez_id()</code> function. If <code>panel</code> and <code>genes</code> are both supplied, <code>genes</code> from both arguments will be returned. If both are NULL (default), it will return gene results for all available genomic data for that sample.   |

panel	One or more panel IDs to query (e.g. 'IMPACT468'). If panel and genes are both supplied, genes from both arguments will be returned. If both are NULL (default), it will return gene results for all available genomic data for that sample.
add_hugo	Logical indicating whether HugoGeneSymbol should be added to your resulting data frame, if not already present in raw API results. Argument is TRUE by default. If FALSE, results will be returned as is (i.e. any existing Hugo Symbol columns in raw results will not be removed).
base_url	The database URL to query If NULL will default to URL set with set_cbiportal_db(<your_db>)

**Value**

A data frame of mutations (maf file format)

**Examples**

```
## Not run:
get_mutations_by_sample(sample_id = c("TCGA-OR-A5J2-01", "TCGA-OR-A5J6-01"),
  study_id = "acc_tcga",
  base_url = "public")

## End(Not run)
```

---

```
get_mutations_by_study
```

*Get Mutations By Study ID*

---

**Description**

Get Mutations By Study ID

**Usage**

```
get_mutations_by_study(
  study_id = NULL,
  molecular_profile_id = NULL,
  add_hugo = TRUE,
  base_url = NULL
)
```

**Arguments**

study_id	A study ID to query mutations. If NULL, guesses study ID based on molecular_profile_id.
molecular_profile_id	a molecular profile to query mutations. If NULL, guesses molecular_profile_id based on study ID.

add_hugo	Logical indicating whether HugoGeneSymbol should be added to your resulting data frame, if not already present in raw API results. Argument is TRUE by default. If FALSE, results will be returned as is (i.e. any existing Hugo Symbol columns in raw results will not be removed).
base_url	The database URL to query If NULL will default to URL set with set_cbioportal_db(<your_db>)

**Value**

A dataframe of mutations (maf file format)

**Examples**

```
## Not run:
get_mutations_by_study(study_id = "prad_msk_2019")
get_mutations_by_study(molecular_profile_id = "prad_msk_2019_mutations")

## End(Not run)
```

---

get\_panel\_by\_sample     *Get Gene Panel by study ID and sample ID*

---

**Description**

Get Gene Panel by study ID and sample ID

**Usage**

```
get_panel_by_sample(
  study_id = NULL,
  sample_id = NULL,
  sample_study_pairs = NULL,
  base_url = NULL
)
```

**Arguments**

study_id	A string indicating the study ID from which to pull data. If no study ID, will guess the study ID based on your URL and inform. Only 1 study ID can be passed. If mutations/cna from more than 1 study needed, see sample_study_pairs
sample_id	a vector of sample IDs (character)
sample_study_pairs	A dataframe with columns: sample_id, study_id and molecular_profile_id (optional). Variations in capitalization of column names are accepted. This can be used in place of sample_id, study_id, molecular_profile_id arguments above if you need to pull samples from several different studies at once. If passed this will take overwrite sample_id, study_id, molecular_profile_id if also passed.
base_url	The database URL to query If NULL will default to URL set with set_cbioportal_db(<your_db>)

**Value**

a dataframe of a specific clinical attribute

**Examples**

```
## Not run:
get_panel_by_sample(study_id = "blca_plasmacytoid_mskcc_2016",
  sample_id = "DS-sig-010-P2",
  base_url = 'www.cbioportal.org/api')

## End(Not run)
```

---

get\_samples\_by\_patient

*Get sample IDs for a given set of patient IDs*

---

**Description**

Get sample IDs for a given set of patient IDs

**Usage**

```
get_samples_by_patient(patient_id = NULL, study_id = NULL, base_url = NULL)
```

**Arguments**

patient_id	A character string of sample IDs to query
study_id	A character string indicating which study ID should be searched. Only 1 study allowed. If NULL, we will guess a default study ID based on your database URL.
base_url	The database URL to query If NULL will default to URL set with set_cbioportal_db(<your_db>)

**Value**

A dataframe of patient IDs and corresponding sample IDs. If patient has multiple samples, there will be multiple rows per patient.

**Examples**

```
## Not run:
get_samples_by_patient(patient_id = c("P-0000034", "P-0000036"))

## End(Not run)
```



---

get_study_info	<i>Get Metadata on All Available Studies in Database or a Specified Study</i>
----------------	---

---

**Description**

Get Metadata on All Available Studies in Database or a Specified Study

**Usage**

```
get_study_info(study_id = NULL, base_url = NULL)
```

**Arguments**

study_id	one or more study IDs (see available_studies() to lookup IDs)
base_url	The database URL to query. If NULL will default to URL set with set_cbioportal_db(<your_db>)

**Value**

A dataframe of study metadata

**Examples**

```
## Not run:
set_cbioportal_db("public")
get_study_info("acc_tcga")

## End(Not run)
```

---

impact_gene_info	<i>IMPACT Gene Meta Data</i>
------------------	------------------------------

---

**Description**

Dataframe labeling all genes included in IMPACT panels along with their corresponding platform ID and Entrez ID.

**Usage**

```
impact_gene_info
```

**Format**

A data frame with 470 genes

**hugo\_symbol** Factor w/ 574 levels, Column containing all HUGO symbols genes included in IMPACT

**entrez\_id** Integer, contains all Entrez IDs for genes included in IMPACT

**platform\_341** Character, indicates whether each gene was included in IMPACT platform 341. Options are included and not included

**platform\_410** Character, indicates whether each gene was included in IMPACT platform 410. Options are included and not included

**platform\_468** Character, indicates whether each gene was included in IMPACT platform 468. Options are included and not included

**alias** A nested dataframe of aliases for each gene and corresponding entrez gene ids for aliases if they exist

**Source**

<http://www.cbioportal.org/>

---

set_cbioportal_db	<i>Connect to cBioPortal DB</i>
-------------------	---------------------------------

---

**Description**

This function sets a base cBioPortal URL

**Usage**

```
set_cbioportal_db(db = NULL)
```

**Arguments**

**db** The database URL to use as base URL for calls, or "public" for <https://www.cbioportal.org/>

**Value**

No return value, called for side effects. Will display an alert notifying if the user has successfully authenticated to cBioPortal.

**Author(s)**

Karissa Whiting, Daniel D. Sjoberg

### Examples

```
## Not run:  
set_cbioportal_db(db = "public")  
  
## End(Not run)
```

---

test\_cbioportal\_db      *Test the Database Connection Anytime During your R Session*

---

### Description

Helps troubleshoot API issues during an R session

### Usage

```
test_cbioportal_db()
```

### Value

No return value, called for side effects. Will display an alert notifying if the user has successfully authenticated to cBioPortal

### Author(s)

Karissa Whiting, Daniel D. Sjoberg

### Examples

```
## Not run:  
set_cbioportal_db("public")  
test_cbioportal_db()  
  
## End(Not run)
```

# Index

## \* datasets

- impact\_gene\_info, [25](#)
  
- available\_clinical\_attributes, [3](#)
- available\_gene\_panels, [3](#)
- available\_patients, [4](#)
- available\_profiles, [5](#)
- available\_samples, [5](#)
- available\_studies, [6](#)
  
- get\_alias, [7](#)
- get\_cbioportal\_token, [7](#)
- get\_clinical\_by\_patient, [8](#)
- get\_clinical\_by\_sample, [9](#)
- get\_clinical\_by\_study, [10](#)
- get\_cna\_by\_sample, [11](#)
- get\_cna\_by\_study, [13](#)
- get\_entrez\_id, [14](#)
- get\_fusions\_by\_sample, [14](#)
- get\_fusions\_by\_study, [16](#)
- get\_gene\_panel, [19](#)
- get\_genes, [17](#)
- get\_genetics\_by\_sample, [17](#)
- get\_genetics\_by\_study, [19](#)
- get\_hugo\_symbol, [20](#)
- get\_mutations\_by\_sample, [21](#)
- get\_mutations\_by\_study, [22](#)
- get\_panel\_by\_sample, [23](#)
- get\_samples\_by\_patient, [24](#)
- get\_structural\_variants\_by\_sample  
    (get\_fusions\_by\_sample), [14](#)
- get\_structural\_variants\_by\_study  
    (get\_fusions\_by\_study), [16](#)
- get\_study\_info, [25](#)
  
- impact\_gene\_info, [25](#)
  
- set\_cbioportal\_db, [26](#)
  
- test\_cbioportal\_db, [27](#)