# Package 'drawProteins'

November 4, 2026

**Title** Package to Draw Protein Schematics from Uniprot API output

**Version** 1.32.0

**Description** This package draws protein schematics from Uniprot API output. From the JSON returned by the GET command, it creates a dataframe from the Uniprot Features API. This dataframe can then be used by geoms based on ggplot2 and base R to draw protein schematics.

**Depends** R (>= 4.0)

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**Imports** ggplot2, httr, dplyr, readr, tidyr

**Suggests** covr, testthat, knitr, rmarkdown, BiocStyle

**VignetteBuilder** knitr

**RoxygenNote** 7.3.3

**URL** https://github.com/brennanpincardiff/drawProteins

**BugReports** https://github.com/brennanpincardiff/drawProteins/issues/new

**biocViews** Visualization, FunctionalPrediction, Proteomics

**git_url** https://git.bioconductor.org/packages/drawProteins

**git_branch** RELEASE_3_23

**git_last_commit** 0ae1012

**git_last_commit_date** 2026-04-28

**Repository** Bioconductor 3.23

**Date/Publication** 2026-11-04

**Author** Paul Brennan [aut, cre]

**Maintainer** Paul Brennan <brennanpincardiff@gmail.com>

# Contents

drawProteins . . . 2
draw_canvas . . . 2
draw_chains . . . 3
draw_domains . . . 4
draw_folding . . . 5
draw_motif . . . 6
draw_phospho . . . 7
draw_recept_dom . . . 8
draw_regions . . . 9
draw_repeat . . . 9
extract_feat_acc . . . 10
extract_names . . . 11
extract_transcripts . . . 12
feature_to_dataframe . . . 13
five_rel_data . . . 13
five_rel_list . . . 14
get_features . . . 15
parse_gff . . . 15
phospho_site_info . . . 16
protein_json . . . 17
rel_A_features . . . 17
rel_json . . . 18
tnfs_data . . . 18

**Index** **20**

---

`drawProteins` *drawProteins.*

---

## Description

This package has been created to allow the visualisation of protein schematics based on the data obtained from the [Uniprot Protein Database](http://www.uniprot.org/).

---

`draw_canvas` *Create ggplot2 object with protein chains from feature database*

---

## Description

`draw_canvas` uses the dataframe containing the protein features to creates the basic plot element by determining the length of the longest protein and the number of proteins to plot.

## Usage

```
draw_canvas(data)
```

### Arguments

| | |
|---|---|
| data | Dataframe of one or more rows with the following column names: 'type', 'description', 'begin', 'end', 'length', 'accession', 'entryName', 'taxid', 'order'. Must contain a minimum of one "CHAIN" as data$type. |

### Value

A ggplot2 object either in the plot window or as an object.

### Examples

```
# draws a blank canvas of the correct size
data("five_rel_data")
draw_canvas(five_rel_data)
```

---

## draw_chains *Create ggplot2 object with protein chains from feature database*

---

### Description

draw_chains uses the dataframe containing the protein features to plot the chains, the full length proteins. It creates the basic plot element by determining the length of the longest protein. The ggplot2 function geom_rect is then used to draw each of the protein chains proportional to their number of amino acids (length).

### Usage

```
draw_chains(p, data = data,
    outline = "black", fill = "grey",
    label_chains = TRUE, labels = data[data$type == "CHAIN",]$entryName,
    linewidth = 0.5, alpha = 1.0, label_size = 4)
```

### Arguments

| | |
|---|---|
| p | ggplot2 object ideally created with draw_canvas. |
| data | Dataframe of one or more rows with the following column names: 'type', 'description', 'begin', 'end', 'length', 'accession', 'entryName', 'taxid', 'order'. Must contain a minimum of one "CHAIN" as data$type. |
| outline | Colour of the outline of each chain. |
| fill | Colour of the fill of each chain. |
| label_chains | Option to label chains or not. |
| labels | Vector with source of names for the chains. EntryName used as default but can be changed. |
| linewidth | Thickness of the outline of the chains. |
| alpha | Transparency of the rectangles representing the chains. |
| label_size | Size of the text used for labels. |

## Value

A ggplot2 object either in the plot window or as an object.

## Examples

```
# combines with draw_canvas to plot and label chains.
data("five_rel_data")
p <- draw_canvas(five_rel_data)
draw_chains(p, five_rel_data)

# draws five chains with different colours to default
data("five_rel_data")
p <- draw_canvas(five_rel_data)
draw_chains(p, five_rel_data,
    label_chains = FALSE,
    fill = "red",
    outline = "grey")
```

---

# draw_domains *Add protein domains to ggplot2 object.*

---

## Description

draw_domains adds domains to the ggplot2 object created by draw_chains. It uses the data object. The ggplot2 function geom_rect is used to draw each of the domain chains proportional to their number of amino acids (length).

## Usage

```
draw_domains(p,
         data = data,
         label_domains = TRUE,
         label_size = 4,
         alpha = 1.0,
         show.legend = TRUE,
         type = "DOMAIN")
```

## Arguments

| | |
|---|---|
| p | ggplot2 object ideally created with draw_canvas. |
| data | Dataframe of one or more rows with the following column names: 'type', 'description', 'begin', 'end', 'length', 'accession', 'entryName', 'taxid', 'order'. Must contain a minimum of one "CHAIN" as data$type. |
| label_domains | Option to label domains or not. |
| label_size | Size of the text used for labels. |
| alpha | Transparency of the rectangles representing the domains. |

| | |
|---|---|
| show.legend | Option to include legend in this layer |
| type | Can change to show other protein features |

## Value

A ggplot2 object either in the plot window or as an object with an additional geom_rect layer.

## Examples

```
# combines with draw_chains to plot chains and domains.
data("five_rel_data")
p <- draw_canvas(five_rel_data)
p <- draw_chains(p, five_rel_data, label_size = 1.25)
draw_domains(p, five_rel_data)
```

---

# draw_folding *Add regions to ggplot object: alpha-helixes, beta-strands and turns.*

---

## Description

draw_folding adds alpha-helixes, beta-strands and turns to the ggplot2 object created by draw_chains. It uses the data object. The ggplot2 function geom_rect is used to draw parts of the protein chain which has alpha-helixes, beta-strands and turns proportional to the number of amino acids (length).

## Usage

```
draw_folding(p, data = data,
show.legend = TRUE,show_strand = TRUE,show_helix = TRUE, show_turn = TRUE,
alpha = 1.0)
```

## Arguments

| | |
|---|---|
| p | ggplot2 object ideally created with draw_canvas. |
| data | Dataframe of one or more rows with the following column names: 'type', 'description', 'begin', 'end', 'length', 'accession', 'entryName', 'taxid', 'order'. Uses STRAND, HELIX and TURN type to indicate these parts of the proteins. |
| show.legend | Option to include legend in this layer |
| show_strand | Option to show STRAND in this layer |
| show_helix | Option to show HELIX in this layer |
| show_turn | Option to show TURN in this layer |
| alpha | alpha Transparency of the rectangles representing the folds. |

## Value

A ggplot2 object either in the plot window or as an object with an additional geom_rect layer.

## Examples

```
# combines with draw_chains to colour chain with helicies, strands and turns.
data("five_rel_data")
p <- draw_canvas(five_rel_data)
p <- draw_chains(p, five_rel_data, label_size = 1.25)
draw_folding(p, five_rel_data)
```

---

`draw_motif` *Add protein motifs sites to ggplot2 object.*

---

## Description

`draw_motif` adds protein motifs from Uniprot to ggplot2 object created by `draw_canvas` and `draw_chains`. It uses the data object. The ggplot2 function `geom_rect` is used to draw each of the motifs proportional to their number of amino acids (length).

## Usage

```
draw_motif(p, data = data, alpha = 1.0, show.legend = TRUE)
```

## Arguments

| | |
|---|---|
| `p` | ggplot2 object ideally created with `draw_canvas`. |
| `data` | Dataframe of one or more rows with the following column names: 'type', 'description', 'begin', 'end', 'length', 'accession', 'entryName', 'taxid', 'order'. Must contain a minimum of one "CHAIN" as data$type. |
| `alpha` | Transparency of the rectangles representing the motifs. |
| `show.legend` | Option to include legend in this layer |

## Value

A ggplot2 object either in the plot window or as an object with an additional geom_rect layer.

## Examples

```
# combines with draw_chains to plot chains and motifs
data("five_rel_data")
p <- draw_canvas(five_rel_data)
p <- draw_chains(p, five_rel_data, label_size = 1.25)
draw_motif(p, five_rel_data)
```

draw_phospho *Add protein phosphorylation sites to ggplot2 object.*

## Description

draw_phospho adds phosphorylation sites to ggplot2 object created by draw_canvas and draw_chains. It uses the data object. The ggplot2 function geom_point is used to draw each of the phosphorylation sites at their location as determined by data object.

## Usage

```
draw_phospho(p, data = data, size = 2,
         fill = "yellow", alpha = 1.0, show.legend = FALSE)
```

## Arguments

| | |
|---|---|
| p | ggplot2 object ideally created with draw_canvas. |
| data | Dataframe of one or more rows with the following column names: 'type', 'description', 'begin', 'end', 'length', 'accession', 'entryName', 'taxid', 'order'. Must contain a minimum of one "CHAIN" as data$type. |
| size | Size of the circle |
| fill | Colour of the circle. |
| alpha | Transparency of the circles representing the protein phosphorylation sites. |
| show.legend | Option to include legend in this layer |

## Value

A ggplot2 object either in the plot window or as an object with an additional geom_point layer.

## Examples

```
# combines will with draw_domains to plot chains and phosphorylation sites.
data("five_rel_data")
p <- draw_canvas(five_rel_data)
p <- draw_chains(p, five_rel_data, label_size = 1.25)
draw_phospho(p, five_rel_data)
```

draw_recept_dom *Add receptor domains to ggplot2 object.*

## Description

draw_recept_dom adds receptor domains to the ggplot2 object created by draw_chains. It uses the data object. The ggplot2 function geom_rect is used to draw each of the domain chains proportional to their number of amino acids (length).

## Usage

```
draw_recept_dom(p, data = data, alpha = 1.0, label_domains = FALSE, label_size = 4,
         show.legend = TRUE)
```

## Arguments

| | |
|---|---|
| p | ggplot2 object ideally created with draw_canvas. |
| data | Dataframe of one or more rows with the following column names: 'type', 'description', 'begin', 'end', 'length', 'accession', 'entryName', 'taxid', 'order'. Uses TOPO_DOM and TRANSMEM type to plot these parts of receptors |
| alpha | Transparency of the rectangles representing the receptor domains. |
| label_domains | Option to label receptor domains or not. |
| label_size | Size of the text used for labels. |
| show.legend | Option to include legend in this layer |

## Value

A ggplot2 object either in the plot window or as an object with an additional geom_rect layer.

## Examples

```
# combines with draw_chains to plot chains and domains.
# we like to draw receptors vertically so flip using ggplot2 functions
# scale_x_reverse and coord_flip
data("tnfs_data")
p <- draw_canvas(tnfs_data)
p <- draw_chains(p, tnfs_data, label_size = 1.25)
draw_recept_dom(p, tnfs_data) + ggplot2::scale_x_reverse() +
ggplot2::coord_flip()
```

## draw_regions *Add protein region sites to ggplot2 object.*

### Description

draw_regions adds protein regions from Uniprot to ggplot2 object created by draw_canvas draw_chains. It uses the data object. The ggplot2 function geom_rect is used to draw each of the regions proportional to their number of amino acids (length).

### Usage

```
draw_regions(p, data = data, alpha = 1.0, show.legend=TRUE)
```

### Arguments

| | |
|---|---|
| p | ggplot2 object ideally created with draw_canvas. |
| data | Dataframe of one or more rows with the following column names: 'type', 'description', 'begin', 'end', 'length', 'accession', 'entryName', 'taxid', 'order'. Must contain a minimum of one "CHAIN" as data$type. |
| alpha | Transparency of the rectangles representing the regions. |
| show.legend | Option to include legend in this layer |

### Value

A ggplot2 object either in the plot window or as an object with an additional geom_rect layer.

### Examples

```
# combines with draw_chains to plot chains and regions.
data("five_rel_data")
p <- draw_canvas(five_rel_data)
p <- draw_chains(p, five_rel_data, label_size = 1.25)
draw_regions(p, five_rel_data)
```

## draw_repeat *Add protein repeats sites to ggplot2 object.*

### Description

draw_repeat adds protein repeats from Uniprot to ggplot2 object created by draw_canvas and draw_chains. It uses the data object. The ggplot2 function geom_rect is used to draw each of the motifs proportional to their number of amino acids (length).

## Usage

```
draw_repeat(p, data = data, label_size = 2, outline = "dimgrey",
            fill = "dimgrey", alpha = 1.0,
            label_repeats = TRUE, show.legend = TRUE)
```

## Arguments

| | |
|---|---|
| p | ggplot2 object ideally created with draw_canvas. |
| data | Dataframe of one or more rows with the following column names: 'type', 'description', 'begin', 'end', 'length', 'accession', 'entryName', 'taxid', 'order'. Must contain a minimum of one "CHAIN" as data$type. |
| label_size | Size of text used for labels of protein repeats. |
| outline | Colour of the outline of each repeat. |
| fill | Colour of the fill of each repeat. |
| alpha | Transparency of the rectangles representing the repeats. |
| label_repeats | Option to label repeats or not. |
| show.legend | Option to include legend in this layer |

## Value

A ggplot2 object either in the plot window or as an object with an additional geom_rect layer.

## Examples

```
# combines with draw_chains to plot chains and repeats.
data("five_rel_data")
p <- draw_canvas(five_rel_data)
p <- draw_chains(p, five_rel_data, label_size = 1.25)
draw_repeat(p, five_rel_data)
```

---

# extract_feat_acc *Create a dataframe of protein features from JSON object (List of 6)*

---

## Description

Converts the list of 6 JSON object created by getting the features from UniProt. Used in the feature_to_dataframe(). Does not give order. Does not operate on List of lists - just the list of 6.

## Usage

```
extract_feat_acc(features_list)
```

### Arguments

| | |
|---|---|
| features_list | A JSON object - list of 6 with features inside. Created as one of the lists in the list of lists by the get_features() function. |

### Value

A dataframe with features: "type", "description", "begin", "end" and adds accession, entryName and taxid for each row.

### Examples

```
data("five_rel_list")
one_protein_features <- extract_feat_acc(five_rel_list[[1]])
head(one_protein_features)
```

---

## extract_names *Extract protein names into a list*

---

### Description

Extracts protein names from JSON object produced by a search of Uniprot with a single protein asking for all the information. The search produces a Large list that contains all the Uniprot information about a protein.

### Usage

```
extract_names(protein_json)
```

### Arguments

| | |
|---|---|
| protein_json | A JSON object from a search with 14 primary parts |

### Value

A List of 6 with "accession", "name", "protein.recommendedName.fullName", gene.name.primary, gene.name.synonym and organism.name.scientific

### Examples

```
# using internal data
data("protein_json")
prot_names <- extract_names(protein_json)
# generates a list of 6

## Not run:
# access the Uniprot Protein API
uniprot_acc <- c("Q04206")  # change this for your fav protein
# Get UniProt entry by accession
```

```
acc_uniprot_url <-
    c("https://www.ebi.ac.uk/proteins/api/proteins?accession=")
comb_acc_api <- paste0(acc_uniprot_url, uniprot_acc)
# basic function is GET() which accesses the API
# requires internet access
protein <- httr::GET(comb_acc_api, accept_json())
status_code(protein)  # returns a 200 means it worked
# use content() function from httr to give us a list
protein_json <- httr::content(protein) # gives a Large list
# with 14 primary parts and lots of bits inside
# function from my package to extract names of protein
names <- extract_names(protein_json)

## End(Not run)
```

---

`extract_transcripts` *Create a new dataframe of protein features from dataframe with multiple transcripts separated so that each transcript is drawn separtely with only the appropriate features.*

---

## Description

This function works on the object returned by the get_features() function. It creates a data.frame of features and includes the accession number AND an order number. It uses the extract_feat_acc function

## Usage

```
extract_transcripts(data)
```

## Arguments

| | |
|---|---|
| `data` | Dataframe of one or more rows with the following column names: 'type', 'description', 'begin', 'end', 'length', 'accession', 'entryName', 'taxid', 'order'. Must contain a minimum of one "CHAIN" as data$type. |

## Value

A dataframe with extra rows if there were multiple transcripts present. Extra transcripts will have an order at the end of the object Each new row should have 9 variables including type, description, begin, end, length, accession, entryName, taxid and order for plotting.

## Examples

```
data(five_rel_data)
new_data <- extract_transcripts(five_rel_data)
# because there are two entries with two transcripts
max(new_data$order) # should now be 7...
```

## feature_to_dataframe *Create a dataframe of protein features from JSON object*

### Description

This function works on the object returned by the get_features() function. It creates a data.frame of features and includes the accession number AND an order number. It uses the extract_feat_acc function below.

### Usage

```
feature_to_dataframe(features_in_lists_of_six)
```

### Arguments

| | |
|---|---|
| features_in_lists_of_six | A list of lists returned by get_features() The number of lists corresponds to the number of accession numbers queried using get_features. The list of 6 contains protein names and features. |

### Value

A dataframe with 9 variables including type, description, begin, end, length, accession, entryName, taxid and order for plotting.

### Examples

```
data("rel_json")
rel_data <- feature_to_dataframe(rel_json)
head(rel_data)

data("five_rel_list")
prot_data <- feature_to_dataframe(five_rel_list)
head(prot_data)
```

## five_rel_data *Dataframe features of 5 human NFkappaB proteins Uniprot on 1 Nov 2017*

### Description

Dataframe features of 5 human NFkappaB proteins Uniprot on 1 Nov 2017

### Usage

```
five_rel_data
```

### Format

A data frame with 320 rows and 9 variables:

**type** type of features - e.g. chain

**description** long name for the protein

**begin** starting position (amino acid number) of feature

**end** ending position (amino acid number) of feature

**length** length of feature - number of amino acids

**accession** protein Uniprot accession number

**entryName** protein Uniprot entry Name

**taxid** taxonomic identification - species

**order** plotting order from the bottom of the graph

### Value

A data frame with 320 rows and 9 variables

### Source

Uniprot http://www.uniprot.org Accession numbers Q04206 Q01201 Q04864 P19838 Q00653

---

## five_rel_list *Features of five human Rel A proteins*

---

### Description

List of features from five human NFkappaB proteins downloaded from Uniprot on 15 August 2017

### Usage

```
five_rel_list
```

### Format

Large List of 5 elements - one element for each protein

### Value

Large List of 5 elements - one element for each protein

### Source

Uniprot http://www.uniprot.org Accession numbers Q04206 Q01201 Q04864 P19838 Q00653

## get_features *GET features of protein(s) from UniProt API*

### Description

This function creates the URL required to query the UniProt API and returns the features of the protein or proteins in JSON format. It uses the GET() function from the httr package.

### Usage

```
get_features(proteins_acc)
```

### Arguments

| | |
|---|---|
| proteins_acc | A vector of length 1 with one or more UniProt accession numbers separated by spaces. |

### Value

If there is internet access and the UniProt accession numbers are good, the function will return a list of lists. The list will be of length equivalent to the number of Uniprot accession numbers supplied. The lists inside will be of length 6 and will contain information about the proteins and the features.

### Examples

```
# Requires internet access
prot_data <- get_features("Q04206 Q01201 Q04864 P19838 Q00653")
```

## parse_gff *Reformat file or url in gff format to allow to draw*

### Description

parse_gff loads a file or downloads from an url if provided protein information that is then changed to allow it to work with draw_canvas and other draw functions in drawProteins.

### Usage

```
parse_gff(file_or_link)
```

### Arguments

| | |
|---|---|
| file_or_link | link in gff format or a file in gff format that can be read by read_tsv function from the readr package. |

## Value

Dataframe of one or more rows with the following column names: 'accession', 'source', 'type', 'begin', 'end', 'order', 'entryName', 'description'. Must contain a minimum of one "CHAIN" as data$type to allow plotting.

## Examples

```
data <- parse_gff("https://www.uniprot.org/uniprot/Q04206.gff")
```

---

# phospho_site_info *Create a dataframe of protein features from JSON object*

---

## Description

Reduces data.frame of features to just phosphorylation sites. Uses a subsetting step and a grep with the pattern "Phospho".

## Usage

```
phospho_site_info(features)
```

## Arguments

| | |
|---|---|
| features | A dataframe of protein features, for example created by the feature_to_dataframe() function. |

## Value

A dataframe that only contains protein phosphorylation sites from Uniprot

## Examples

```
data("five_rel_data")
sites <- phospho_site_info(five_rel_data)
head(sites)
```

## protein_json *Uniprot infor human Rel A protein in JSON format*

### Description

Large list (968.8 Kb) of information about human Rel A downloaded from Uniprot on 1 November 2017

### Usage

```
protein_json
```

### Format

List of 1 with List of 14 inside

### Value

List of 6 - information necessary to draw Rel A/p65

### Source

http://www.uniprot.org/uniprot/Q04206

## rel_A_features *Features of human Rel A protein*

### Description

List of features from human Rel A downloaded from Uniprot on 15 August 2017

### Usage

```
rel_A_features
```

### Format

List of 6 - information necessary to draw Rel A/p65

### Value

List of 6 - information necessary to draw Rel A/p65

### Source

http://www.uniprot.org/uniprot/Q04206

## rel_json — *Human Rel A protein features in JSON format*

### Description

List of 1 with List of 6 inside downloaded from Uniprot on 1 November 2017

### Usage

```
rel_json
```

### Format

List of 1 with List of 6 - information necessary to draw Rel A/p65

### Value

List of 1 with List of 6 - information necessary to draw Rel A/p65

### Source

http://www.uniprot.org/uniprot/Q04206

## tnfs_data — *Dataframe features of 2 human TNF receptors from Uniprot on 3 Jan 2018*

### Description

Dataframe features of 2 human TNF receptors from Uniprot on 3 Jan 2018

### Usage

```
tnfs_data
```

### Format

A data frame with 127 rows of 9 variables:

**type** type of features - e.g. chain
**description** long name for the protein
**begin** starting position (amino acid number) of feature
**end** ending position (amino acid number) of feature
**length** length of feature - number of amino acids
**accession** protein Uniprot accession number
**entryName** protein Uniprot entry Name
**taxid** taxonomic identification - species
**order** plotting order from the bottom of the graph

## Value

A data frame with 127 rows and 9 variables

## Source

Uniprot http://www.uniprot.org Accession numbers P19438 P25942

# Index

∗ **datasets**
five_rel_data, 13
five_rel_list, 14
protein_json, 17
rel_A_features, 17
rel_json, 18
tnfs_data, 18

draw_canvas, 2, *3–10*
draw_chains, 3, *4–9*
draw_domains, 4
draw_folding, 5
draw_motif, 6
draw_phospho, 7
draw_recept_dom, 8
draw_regions, 9
draw_repeat, 9
drawProteins, 2

extract_feat_acc, 10
extract_names, 11
extract_transcripts, 12

feature_to_dataframe, 13
five_rel_data, 13
five_rel_list, 14

geom_point, *7*
get_features, 15

parse_gff, 15
phospho_site_info, 16
protein_json, 17

rel_A_features, 17
rel_json, 18

tnfs_data, 18