MyGene.info R Client

Adam Mark, Ryan Thompson, Chunlei Wu

November 1, 2022

Contents

1 Overview

MyGene.Info provides simple-to-use REST web services to query/retrieve gene annotation data. It's designed with simplicity and performance emphasized. *mygene* is an easy-to-use R wrapper to access MyGene.Info services.

2 Gene Annotation Service

2.1 `getGene`

- Use `getGene`, the wrapper for GET query of "/gene/<geneid>" service, to return the gene object for the given geneid.

```
> gene <- getGene("1017", fields="all")
> length(gene)

[1] 1

> gene["name"]

[[1]]
NULL

> gene["taxid"]

[[1]]
NULL

> gene["uniprot"]

[[1]]
NULL

> gene["refseq"]

[[1]]
NULL
```

2.2 `getGenes`

- Use `getGenes`, the wrapper for POST query of "/gene" service, to return the list of gene objects for the given character vector of geneids.

```
> getGenes(c("1017","1018","ENSG00000148795"))

DataFrame with 3 rows and 8 columns
```

```
             query         _id X_version  entrezgene                  name
       <character> <character> <integer> <character>           <character>
1            1017        1017         3        1017 cyclin dependent kin..
2            1018        1018         2        1018 cyclin dependent kin..
3 ENSG00000148795          NA        NA          NA                    NA
       symbol     taxid  notfound
  <character> <integer> <logical>
1        CDK2      9606        NA
2        CDK3      9606        NA
3          NA        NA      TRUE
```

3 Gene Query Service

3.1 `query`

- Use `query`, a wrapper for GET query of "/query?q=<query>" service, to return the query result.

```
> query(q="cdk2", size=5)

$took
[1] 85

$total
[1] 1200

$max_score
[1] 87.15067

$hits
                 _id   _score entrezgene                     name     symbol
1               1017 87.15067       1017 cyclin dependent kinase 2       CDK2
2    ENSG00000123374 87.15067       <NA> cyclin dependent kinase 2       CDK2
3 ENSMUSG00000025358 73.09411       <NA> cyclin-dependent kinase 2       Cdk2
4              12566 73.09411      12566 cyclin-dependent kinase 2       Cdk2
5            5880545 62.88823    5880545                     CDK2 EDI_169580
   taxid
1   9606
2   9606
3  10090
4  10090
```

```
5 370354
```

```
> query(q="NM_013993")

$took
[1] 13

$total
[1] 1

$max_score
[1] 3.782039

$hits
  _id   _score entrezgene                                        name symbol
1 780 3.782039        780 discoidin domain receptor tyrosine kinase 1   DDR1
  taxid
1  9606
```

3.2 queryMany

- Use `queryMany`, a wrapper for POST query of "/query" service, to return the batch query result.

```
> queryMany(c('1053_at', '117_at', '121_at', '1255_g_at', '1294_at'),
+           scopes="reporter", species="human")

Finished
Pass returnall=TRUE to return lists of duplicate or missing query terms.
DataFrame with 6 rows and 7 columns
        query         _id   X_score  entrezgene                  name
  <character> <character> <numeric> <character>           <character>
1     1053_at        5982   19.8026        5982 replication factor C..
2      117_at        3310   19.1973        3310 heat shock protein f..
3      121_at        7849   19.8809        7849           paired box 8
4   1255_g_at        2978   19.8868        2978 guanylate cyclase ac..
5     1294_at        7318   18.7388        7318 ubiquitin like modif..
6     1294_at   100847079   18.7388   100847079          microRNA 5193
       symbol     taxid
  <character> <integer>
1        RFC2      9606
2       HSPA6      9606
```

```
3        PAX8      9606
4      GUCA1A      9606
5        UBA7      9606
6     MIR5193      9606
```

4 makeTxDbFromMyGene

TxDb is a container for storing transcript annotations. makeTxDbFromMyGene allows the user to make a TxDb object in the Genomic Features package from a mygene "exons" query using a default mygene object.

```
> xli <- c('CCDC83',
+        'MAST3',
+        'RPL11',
+        'ZDHHC20',
+        'LUC7L3',
+        'SNORD49A',
+        'CTSH',
+        'ACOT8')
> txdb <- makeTxDbFromMyGene(xli,
+           scopes="symbol", species="human")
> transcripts(txdb)

GRanges object with 17 ranges and 2 metadata columns:
       seqnames            ranges strand |     tx_id      tx_name
          <Rle>         <IRanges>  <Rle> | <integer>  <character>
   [1]       11 85855382-85920013      + |         1 NM_001286159
   [2]       11 85855382-85920013      + |         2    NM_173556
   [3]       19 18097777-18151686      + |         3    NM_015016
   [4]        1 23691805-23696835      + |         4    NM_000975
   [5]        1 23691778-23696426      + |         5 NM_001199802
   ...      ...               ...    ... .       ...          ...
  [13]       17 50719602-50756215      + |        13    NM_016424
  [14]       17 16440035-16440106      + |        14    NR_002744
  [15]       15 78921749-78945098      - |        15 NM_001319137
  [16]       15 78921059-78945046      - |        16    NM_004390
  [17]       20 45841720-45857392      - |        17    NM_005469
  -------
  seqinfo: 7 sequences from an unspecified genome; no seqlengths
```

makeTxDbFromMyGene invokes either the query or queryMany method and passes the response to construct a `TxDb` object. See `?TxDb` for methods to utilize and access transcript annotations.

5 Tutorial, ID mapping

ID mapping is a very common, often not fun, task for every bioinformatician. Supposedly you have a list of gene symbols or reporter ids from an upstream analysis, and then your next analysis requires to use gene ids (e.g. Entrez gene ids or Ensembl gene ids). So you want to convert that list of gene symbols or reporter ids to corresponding gene ids.

Here we want to show you how to do ID mapping quickly and easily.

5.1 Mapping gene symbols to Entrez gene ids

Suppose xli is a list of gene symbols you want to convert to entrez gene ids:

```
> xli <- c('DDX26B',
+        'CCDC83',
+        'MAST3',
+        'FLOT1',
+        'RPL11',
+        'ZDHHC20',
+        'LUC7L3',
+        'SNORD49A',
+        'CTSH',
+        'ACOT8')
```

You can then call `queryMany` method, telling it your input is `symbol`, and you want `entrezgene` (Entrez gene ids) back.

```
> queryMany(xli, scopes="symbol", fields="entrezgene", species="human")

Finished
Pass returnall=TRUE to return lists of duplicate or missing query terms.
DataFrame with 24 rows and 5 columns
          query  notfound             _id   X_score  entrezgene
    <character> <logical>     <character> <numeric> <character>
1        DDX26B      TRUE              NA        NA          NA
2        CCDC83        NA ENSG00000150676   17.7161          NA
3        CCDC83        NA          220047   17.7161      220047
4         MAST3        NA ENSG00000099308   17.8730          NA
```

```
5         MAST3        NA           23031   17.8730       23031
...         ...       ...             ...       ...         ...
20     SNORD49A        NA           26800   22.1535       26800
21         CTSH        NA ENSG00000103811   17.3592          NA
22         CTSH        NA            1512   17.3592        1512
23        ACOT8        NA ENSG00000101473   17.3210          NA
24        ACOT8        NA           10005   17.3210       10005
```

5.2 Mapping gene symbols to Ensembl gene ids

Now if you want Ensembl gene ids back:

```
> out <- queryMany(xli, scopes="symbol", fields="ensembl.gene", species="human")

Finished
Pass returnall=TRUE to return lists of duplicate or missing query terms.

> out

DataFrame with 24 rows and 4 columns
          query  notfound             _id   X_score
    <character> <logical>     <character> <numeric>
1        DDX26B      TRUE              NA        NA
2        CCDC83        NA ENSG00000150676   17.7095
3        CCDC83        NA          220047   17.7095
4         MAST3        NA ENSG00000099308   17.8823
5         MAST3        NA           23031   17.8823
...         ...       ...             ...       ...
20     SNORD49A        NA           26800   22.1580
21         CTSH        NA ENSG00000103811   17.3610
22         CTSH        NA            1512   17.3610
23        ACOT8        NA ENSG00000101473   17.3319
24        ACOT8        NA           10005   17.3319

> out$ensembl[[4]]$gene

NULL
```

5.3 When an input has no matching gene

In case that an input id has no matching gene, you will be notified from the output.The returned list for this query term contains `notfound` value as True.

```
> xli <- c('DDX26B',
+        'CCDC83',
+        'MAST3',
+        'FLOT1',
+        'RPL11',
+        'Gm10494')
> queryMany(xli, scopes="symbol", fields="entrezgene", species="human")
```

```
Finished
Pass returnall=TRUE to return lists of duplicate or missing query terms.
DataFrame with 16 rows and 5 columns
          query  notfound             _id   X_score  entrezgene
    <character> <logical>     <character> <numeric> <character>
1        DDX26B      TRUE              NA        NA          NA
2        CCDC83        NA          220047   17.7044      220047
3        CCDC83        NA ENSG00000150676   17.7044          NA
4         MAST3        NA ENSG00000099308   17.8823          NA
5         MAST3        NA           23031   17.8823       23031
...         ...       ...             ...       ...         ...
12        FLOT1        NA ENSG00000232280   18.0489          NA
13        FLOT1        NA ENSG00000236271   18.0489          NA
14        RPL11        NA            6135   16.5316        6135
15        RPL11        NA ENSG00000142676   16.5316          NA
16      Gm10494      TRUE              NA        NA          NA
```

5.4 When input ids are not just symbols

```
> xli <- c('DDX26B',
+        'CCDC83',
+        'MAST3',
+        'FLOT1',
+        'RPL11',
+        'Gm10494',
+        '1007_s_at',
+        'AK125780')
>
```

Above id list contains symbols, reporters and accession numbers, and supposedly we want to get back both Entrez gene ids and uniprot ids. Parameters `scopes`, `fields`, `species` are all flexible enough to support multiple values, either a list or a comma-separated string:

```
> out <- queryMany(xli, scopes=c("symbol", "reporter","accession"),
+              fields=c("entrezgene","uniprot"), species="human")

Finished
Pass returnall=TRUE to return lists of duplicate or missing query terms.

> out

DataFrame with 19 rows and 7 columns
          query  notfound             _id   X_score  entrezgene
    <character> <logical>     <character> <numeric> <character>
1        DDX26B      TRUE              NA        NA          NA
2        CCDC83        NA ENSG00000150676   17.7110          NA
3        CCDC83        NA          220047   17.7110      220047
4         MAST3        NA ENSG00000099308   17.8823          NA
5         MAST3        NA           23031   17.8823       23031
...         ...       ...             ...       ...         ...
15        RPL11        NA ENSG00000142676   16.5326          NA
16      Gm10494      TRUE              NA        NA          NA
17    1007_s_at        NA       100616237   18.3334   100616237
18    1007_s_at        NA             780   18.3334         780
19     AK125780        NA       118142757   17.5746   118142757
    uniprot.Swiss.Prot                         uniprot.TrEMBL
           <character>                                 <list>
1                   NA
2                   NA
3               Q8IWF9                              H0YDV3
4                   NA
5               O60307                   A0A8I5KST9,V9GYV0
...                ...                                 ...
15                  NA
16                  NA
17                  NA
18              Q08345 A0A024RCJ0,A0A024RCL1,A0A0A0MSX3,...
19              P43080                   A0A7I2V6E2,B2R9P6

> out$uniprot.Swiss.Prot[[5]]

[1] "O60307"
```

5.5 When an input id has multiple matching genes

From the previous result, you may have noticed that query term `1007_s_at` matches two genes. In that case, you will be notified from the output, and the returned result will include both matching genes.

By passing `returnall=TRUE`, you will get both duplicate or missing query terms

```
> queryMany(xli, scopes=c("symbol", "reporter", "accession"),
+           fields=c("entrezgene", "uniprot"), species='human', returnall=TRUE)

Finished
$response
DataFrame with 19 rows and 7 columns
          query  notfound             _id   X_score  entrezgene
    <character> <logical>     <character> <numeric> <character>
1        DDX26B      TRUE              NA        NA          NA
2        CCDC83        NA ENSG00000150676   17.7178          NA
3        CCDC83        NA          220047   17.7178      220047
4         MAST3        NA           23031   17.8841       23031
5         MAST3        NA ENSG00000099308   17.8841          NA
...         ...       ...             ...       ...         ...
15        RPL11        NA ENSG00000142676   16.5326          NA
16      Gm10494      TRUE              NA        NA          NA
17    1007_s_at        NA       100616237   18.6329   100616237
18    1007_s_at        NA             780   18.6329         780
19     AK125780        NA       118142757   17.5779   118142757
    uniprot.Swiss.Prot                         uniprot.TrEMBL
           <character>                                 <list>
1                   NA
2                   NA
3               Q8IWF9                              H0YDV3
4               O60307                   A0A8I5KST9,V9GYV0
5                   NA
...                ...                                 ...
15                  NA
16                  NA
17                  NA
18              Q08345 A0A024RCJ0,A0A024RCL1,A0A0A0MSX3,...
19              P43080                   A0A7I2V6E2,B2R9P6

$duplicates
  X1007_s_at CCDC83 FLOT1 MAST3 RPL11
1          2      2     8     2     2
```

```
$missing
[1] "DDX26B"  "Gm10494"
```

The returned result above contains `out` for mapping output, `missing` for missing query terms (a list), and `dup` for query terms with multiple matches (including the number of matches).

5.6 Can I convert a very large list of ids?

Yes, you can. If you pass an id list (i.e., `xli` above) larger than 1000 ids, we will do the id mapping in-batch with 1000 ids at a time, and then concatenate the results all together for you. So, from the user-end, it's exactly the same as passing a shorter list. You don't need to worry about saturating our backend servers. Large lists, however, may take a while longer to query, so please wait patiently.

6 References

Wu C, MacLeod I, Su AI (2013) BioGPS and MyGene.info: organizing online, gene-centric information. Nucl. Acids Res. 41(D1): D561-D565. help@mygene.info