

MyGene.info R Client

Adam Mark, Ryan Thompson, Chunlei Wu

April 15, 2025

Contents

1	Overview	2
2	Gene Annotation Service	2
2.1	<code>getGene</code>	2
2.2	<code>getGenes</code>	2
3	Gene Query Service	3
3.1	<code>query</code>	3
3.2	<code>queryMany</code>	4
4	<code>makeTxDbFromMyGene</code>	5
5	Tutorial, ID mapping	6
5.1	Mapping gene symbols to Entrez gene ids	6
5.2	Mapping gene symbols to Ensembl gene ids	7
5.3	When an input has no matching gene	8
5.4	When input ids are not just symbols	8
5.5	When an input id has multiple matching genes	10
5.6	Can I convert a very large list of ids?	11
6	References	11

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 7 columns
```

```

      query      _id X_version  entrezgene      name
<character> <character> <integer> <character> <character>
1      1017      1017      1      1017 cyclin dependent kin..
2      1018      1018      1      1018 cyclin dependent kin..
3 ENSG00000148795      1586      1      1586 cytochrome P450 fami..
      symbol      taxid
<character> <integer>
1      CDK2      9606
2      CDK3      9606
3      CYP17A1      9606

```

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] 15

$total
[1] 2016

$max_score
[1] 139.2848

$hits
      _id      _score  entrezgene      name symbol
1      1017 139.28480      1017 cyclin dependent kinase 2  CDK2
2      12566 116.81951      12566 cyclin dependent kinase 2  Cdk2
3      362817 98.84729      362817 cyclin dependent kinase 2  Cdk2
4 ENSOPRG00000007864 89.86117      <NA> cyclin dependent kinase 2  CDK2
5      114814505 89.86117      114814505 cyclin dependent kinase 2  CDK2
      taxid
1      9606
2      10090
3      10116
4      9978

```

```
5 9258
```

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

$took
[1] 3

$total
[1] 1

$max_score
[1] 1.706854

$hits
  _id  _score entrezgene          name symbol
1 780 1.706854      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  19.0125     5982 replication factor C..
2 117_at     3310  19.0125     3310 heat shock protein f..
3 121_at     7849  19.0125     7849          paired box 8
4 1255_g_at  2978  19.0125     2978 guanylate cyclase ac..
5 1294_at  100847079  17.9989    100847079          microRNA 5193
6 1294_at     7318  17.9989     7318 ubiquitin like modif..
  symbol      taxid
  <character> <integer>
1   RFC2       9606
2  HSPA6       9606
```

```

3      PAX8      9606
4      GUCA1A   9606
5      MIR5193  9606
6      UBA7     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 10 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	17.9630	220047
3	MAST3	NA	23031	18.1863	23031
4	FLOT1	NA	10211	18.5218	10211

MyGene.info R Client

5	RPL11	NA	6135	16.3451	6135
6	ZDHHC20	NA	253832	18.1771	253832
7	LUC7L3	NA	51747	17.6591	51747
8	SNORD49A	NA	26800	23.6848	26800
9	CTSH	NA	1512	17.6223	1512
10	ACOT8	NA	10005	17.6308	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 10 rows and 5 columns
  query notfound _id X_score
<character> <logical> <character> <numeric>
1 DDX26B TRUE NA NA
2 CCDC83 NA 220047 17.9630
3 MAST3 NA 23031 18.1863
4 FLOT1 NA 10211 18.5218
5 RPL11 NA 6135 16.3451
6 ZDHHC20 NA 253832 18.1771
7 LUC7L3 NA 51747 17.6591
8 SNORD49A NA 26800 23.6848
9 CTSH NA 1512 17.6223
10 ACOT8 NA 10005 17.6308

                                ensembl
                                <list>
1
2                                ENSG00000150676
3                                ENSG00000099308
4 ENSG00000230143,ENSG00000224740,ENSG00000137312
5                                ENSG00000142676
6                                ENSG00000180776
7                                ENSG00000108848
8                                ENSG00000277370
9                                ENSG00000103811
10                               ENSG00000101473
```

MyGene.info R Client

```
> out$ensembl[[4]]$gene
[1] "ENSG00000230143" "ENSG00000224740" "ENSG00000137312" "ENSG00000232280"
[5] "ENSG00000206480" "ENSG00000236271" "ENSG00000223654" "ENSG00000206379"
```

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 6 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	17.9630	220047
3	MAST3	NA	23031	18.1863	23031
4	FLOT1	NA	10211	18.5218	10211
5	RPL11	NA	6135	16.3451	6135
6	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')
>
```


MyGene.info R Client

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 9 rows and 7 columns

	query	notfound	_id	X_score	entrezgene	uniprot.Swiss.Prot
	<character>	<logical>	<character>	<numeric>	<character>	<character>
1	DDX26B	TRUE	NA	NA	NA	NA
2	CCDC83	NA	220047	17.9630	220047	Q8IWF9
3	MAST3	NA	23031	18.1863	23031	060307
4	FL0T1	NA	10211	18.5218	10211	075955
5	RPL11	NA	6135	16.3451	6135	P62913
6	Gm10494	TRUE	NA	NA	NA	NA
7	1007_s_at	NA	100616237	17.9989	100616237	NA
8	1007_s_at	NA	780	17.9989	780	Q08345
9	AK125780	NA	118142757	21.1195	118142757	P43080

uniprot.TrEMBL

<list>

1

2

H0YDV3

3 A0A8V8TLL8, A0A994J700, V9GYV0, ...

4 A2AB11, Q5ST80, A2AB10, ...

5 Q5VVC8, Q5VVD0, A0A2R8Y447

6

7

8 A0A024RCQ1, A0A0A0MSX3, Q96T61, ...

9 A0A7I2V6E2, B2R9P6

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

```
[1] "P62913"
```

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 9 rows and 7 columns
  query   notfound   _id  X_score  entrezgene  uniprot.Swiss.Prot
<character> <logical> <character> <numeric> <character> <character>
1   DDX26B     TRUE      NA      NA         NA             NA
2   CCDC83     NA      220047  17.9630    220047        Q8IWF9
3   MAST3      NA      23031  18.1863    23031        060307
4   FL0T1      NA      10211  18.5218    10211        075955
5   RPL11      NA      6135   16.3451    6135         P62913
6   Gm10494    TRUE      NA      NA         NA             NA
7  1007_s_at   NA      100616237  17.9989    100616237     NA
8  1007_s_at   NA      780     17.9989    780           Q08345
9  AK125780    NA      118142757  21.1195    118142757     P43080
  uniprot.TrEMBL
  <list>
1
2
3   H0YDV3
4  A0A8V8TLL8, A0A994J700, V9GYV0, ...
5  A2AB11, Q5ST80, A2AB10, ...
6  Q5VVC8, Q5VVD0, A0A2R8Y447
7
8  A0A024RCQ1, A0A0A0MSX3, Q96T61, ...
9  A0A7I2V6E2, B2R9P6

$duplicates
  X1007_s_at
1         2

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

MyGene.info R Client

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