

MyGene.info R Client

Adam Mark, Ryan Thompson, Chunlei Wu

October 14, 2015

Contents

| | | |
|----------|--------------------------------------------------------|----------|
| 1 | Overview | 1 |
| 2 | Gene Annotation Service | 2 |
| 2.1 | getGene | 2 |
| 2.2 | getGenes | 2 |
| 3 | Gene Query Service | 3 |
| 3.1 | query | 3 |
| 3.2 | queryMany | 4 |
| 4 | makeTxDbFromMyGene | 4 |
| 5 | Tutorial, ID mapping | 5 |
| 5.1 | Mapping gene symbols to Entrez gene ids | 5 |
| 5.2 | Mapping gene symbols to Ensembl gene ids | 6 |
| 5.3 | When an input has no matching gene | 7 |
| 5.4 | When input ids are not just symbols | 7 |
| 5.5 | When an input id has multiple matching genes | 8 |
| 5.6 | Can I convert a very large list of ids? | 9 |
| 6 | References | 9 |

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] 37
> gene$name
[1] "cyclin-dependent kinase 2"
> gene$taxid
[1] 9606
> gene$uniprot
$TrEMBL
[1] "AOA024RB10" "AOA024RB77" "B4DDL9"      "E7ESI2"      "G3V317"      "G3V5T9"

$`Swiss-Prot`
[1] "P24941"

> gene$refseq
$protein
[1] "NP_001277159" "NP_001789"      "NP_439892"      "XP_011536034"

$genomic
[1] "NC_000012"      "NC_018923"      "NG_034014"      "NT_029419"      "NW_004929384"

$rna
[1] "NM_001290230" "NM_001798"      "NM_052827"      "XM_011537732"
```

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 6 columns
      symbol      query      name
<character> <character> <character>
1      CDK2      1017      cyclin-dependent kinase 2
```

```

2      CDK3      1018      cyclin-dependent kinase 3
3      CYP17A1 ENSG00000148795 cytochrome P450, family 17, subfamily A, polypeptide 1
  entrezgene      _id      taxid
  <integer> <character> <integer>
1      1017      1017      9606
2      1018      1018      9606
3      1586      1586      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)
```

```
$hits
```

| | symbol | taxid | name | _score | entrezgene | _id |
|---|---------|-------|----------------------------------|-----------|------------|--------|
| 1 | CDK2 | 9606 | cyclin-dependent kinase 2 | 368.29034 | 1017 | 1017 |
| 2 | Cdk2 | 10090 | cyclin-dependent kinase 2 | 347.59430 | 12566 | 12566 |
| 3 | Cdk2 | 10116 | cyclin dependent kinase 2 | 261.36737 | 362817 | 362817 |
| 4 | Cdk2ap2 | 10090 | CDK2-associated protein 2 | 21.80851 | 52004 | 52004 |
| 5 | CACUL1 | 9606 | CDK2-associated, cullin domain 1 | 19.95450 | 143384 | 143384 |

```
$max_score
```

```
[1] 368.2903
```

```
$took
```

```
[1] 3
```

```
$total
```

```
[1] 22
```

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

```
$hits
```

| | entrezgene | symbol | _id | _score | name | taxid |
|---|------------|--------|-----|-----------|---------------------------------------------|-------|
| 1 | 780 | DDR1 | 780 | 0.5173506 | discoidin domain receptor tyrosine kinase 1 | 9606 |

```
$max_score
```

```
[1] 0.5173506
```

```
$took
```

```
[1] 3
```

```
$total
[1] 1
```

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

DataFrame with 5 rows and 6 columns

| | taxid | _id | query | entrezgene | | |
|---|-----------|-------------|-------------|------------|---------------------------------------------|-------------|
| | <integer> | <character> | <character> | <integer> | | |
| 1 | 9606 | 5982 | 1053_at | 5982 | | |
| 2 | 9606 | 3310 | 117_at | 3310 | | |
| 3 | 9606 | 7849 | 121_at | 7849 | | |
| 4 | 9606 | 2978 | 1255_g_at | 2978 | | |
| 5 | 9606 | 7318 | 1294_at | 7318 | | |
| | | | | | name | symbol |
| | | | | | <character> | <character> |
| 1 | | | | | replication factor C (activator 1) 2, 40kDa | RFC2 |
| 2 | | | | | heat shock 70kDa protein 6 (HSP70B') | HSPA6 |
| 3 | | | | | paired box 8 | PAX8 |
| 4 | | | | | guanylate cyclase activator 1A (retina) | GUCA1A |
| 5 | | | | | ubiquitin-like modifier activating enzyme 7 | UBA7 |

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

```
+           scopes="symbol", species="human")
> transcripts(txdb)
```

GRanges object with 15 ranges and 2 metadata columns:

| | seqnames | ranges | strand | tx_id | tx_name |
|------|----------|------------------------|--------|-----------|--------------|
| | <Rle> | <IRanges> | <Rle> | <integer> | <character> |
| [1] | X | [135520629, 135582535] | + | 1 | NM_182540 |
| [2] | 11 | [85855100, 85920020] | + | 2 | NM_173556 |
| [3] | 11 | [85855100, 85920020] | + | 3 | NM_001286159 |
| [4] | 19 | [18097792, 18151689] | + | 4 | NM_015016 |
| [5] | 1 | [23691778, 23696426] | + | 5 | NM_001199802 |
| ... | ... | ... | ... | ... | ... |
| [11] | 17 | [50719564, 50752711] | + | 11 | NM_016424 |
| [12] | 17 | [50719564, 50752711] | + | 12 | NM_006107 |
| [13] | 17 | [16440035, 16440106] | + | 13 | NR_002744 |
| [14] | 15 | [78921749, 78945078] | - | 14 | NM_004390 |
| [15] | 20 | [45841720, 45857409] | - | 15 | NM_005469 |

seqinfo: 8 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

DataFrame with 10 rows and 3 columns

| | entrezgene | query | _id |
|----|------------|-------------|-------------|
| | <integer> | <character> | <character> |
| 1 | 203522 | DDX26B | 203522 |
| 2 | 220047 | CCDC83 | 220047 |
| 3 | 23031 | MAST3 | 23031 |
| 4 | 10211 | FLOT1 | 10211 |
| 5 | 6135 | RPL11 | 6135 |
| 6 | 253832 | ZDHHC20 | 253832 |
| 7 | 51747 | LUC7L3 | 51747 |
| 8 | 26800 | SNORD49A | 26800 |
| 9 | 1512 | CTSH | 1512 |
| 10 | 10005 | ACOT8 | 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

```
> out
```

DataFrame with 10 rows and 4 columns

| | ensembl.gene | query | _id | ensembl |
|----|-----------------|-------------|-------------|---------|
| | <character> | <character> | <character> | <List> |
| 1 | ENSG00000165359 | DDX26B | 203522 | ##### |
| 2 | ENSG00000150676 | CCDC83 | 220047 | ##### |
| 3 | ENSG00000099308 | MAST3 | 23031 | ##### |
| 4 | NA | FLOT1 | 10211 | ##### |
| 5 | ENSG00000142676 | RPL11 | 6135 | ##### |
| 6 | ENSG00000180776 | ZDHHC20 | 253832 | ##### |
| 7 | ENSG00000108848 | LUC7L3 | 51747 | ##### |
| 8 | ENSG00000277370 | SNORD49A | 26800 | ##### |
| 9 | ENSG00000103811 | CTSH | 1512 | ##### |
| 10 | ENSG00000101473 | ACOT8 | 10005 | ##### |

```
> out$ensembl.gene[[4]]
```

```
[1] NA
```

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 4 columns

| | query | entrezgene | _id | notfound |
|---|-------------|------------|-------------|-----------|
| | <character> | <integer> | <character> | <logical> |
| 1 | DDX26B | 203522 | 203522 | NA |
| 2 | CCDC83 | 220047 | 220047 | NA |
| 3 | MAST3 | 23031 | 23031 | NA |
| 4 | FLOT1 | 10211 | 10211 | NA |
| 5 | RPL11 | 6135 | 6135 | NA |
| 6 | Gm10494 | NA | NA | TRUE |

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 9 rows and 6 columns

| | <code>_id</code> | <code>entrezgene</code> | <code>uniprot.Swiss-Prot</code> | <code>uniprot.TrEMBL</code> | <code>query</code> | <code>notfound</code> |
|---|------------------|-------------------------|---------------------------------|-----------------------------|--------------------|-----------------------|
| | <character> | <integer> | <character> | <List> | <character> | <logical> |
| 1 | 203522 | 203522 | Q5JSJ4 | ##### | DDX26B | NA |
| 2 | 220047 | 220047 | Q8IWF9 | ##### | CCDC83 | NA |
| 3 | 23031 | 23031 | O60307 | ##### | MAST3 | NA |
| 4 | 10211 | 10211 | O75955 | ##### | FLOT1 | NA |
| 5 | 6135 | 6135 | P62913 | ##### | RPL11 | NA |
| 6 | NA | NA | NA | ##### | Gm10494 | TRUE |
| 7 | 100616237 | 100616237 | NA | ##### | 1007_s_at | NA |
| 8 | 780 | 780 | Q08345 | ##### | 1007_s_at | NA |
| 9 | 2978 | 2978 | P43080 | ##### | AK125780 | NA |

```
> 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 6 columns

| | <code>uniprot.Swiss-Prot</code> | <code>uniprot.TrEMBL</code> | <code>_id</code> | <code>query</code> | <code>entrezgene</code> | <code>notfound</code> |
|---|---------------------------------|-----------------------------|------------------|--------------------|-------------------------|-----------------------|
| | <character> | <List> | <character> | <character> | <integer> | <logical> |
| 1 | Q5JSJ4 | ##### | 203522 | DDX26B | 203522 | NA |
| 2 | Q8IWF9 | ##### | 220047 | CCDC83 | 220047 | NA |
| 3 | O60307 | ##### | 23031 | MAST3 | 23031 | NA |
| 4 | O75955 | ##### | 10211 | FLOT1 | 10211 | NA |
| 5 | P62913 | ##### | 6135 | RPL11 | 6135 | NA |
| 6 | NA | ##### | NA | Gm10494 | NA | TRUE |
| 7 | NA | ##### | 100616237 | 1007_s_at | 100616237 | NA |
| 8 | Q08345 | ##### | 780 | 1007_s_at | 780 | NA |
| 9 | P43080 | ##### | 2978 | AK125780 | 2978 | NA |


```
$duplicates
  X1007_s_at
1           2
```

```
$missing
[1] "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