MyGene.info R Client

Adam Mark, Ryan Thompson, Chunlei Wu

September 26, 2014

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] 36

> gene$name

[1] "cyclin-dependent kinase 2"

> gene$taxid

[1] 9606

> gene$uniprot

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

$TrEMBL
[1] "B4DDL9" "E7ESI2" "G3V317" "G3V5T9"

> gene$refseq

$genomic
[1] "AC_000144"    "NC_000012"    "NC_018923"    "NG_034014"    "NT_029419"   
[6] "NW_001838059" "NW_004929384"

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

$protein
[1] "NP_001277159" "NP_001789"    "NP_439892"   
```

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"))

 chr "symbol,name,taxid,entrezgene"
DataFrame with 3 rows and 6 columns
                                                    name      symbol     taxid entrezgene
```

```
                                             <character> <character> <numeric>  <numeric>
1                              cyclin-dependent kinase 2        CDK2      9606       1017
2                              cyclin-dependent kinase 3        CDK3      9606       1018
3 cytochrome P450, family 17, subfamily A, polypeptide 1     CYP17A1      9606       1586
            query         _id
      <character> <character>
1            1017        1017
2            1018        1018
3 ENSG00000148795        1586
```

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
  entrezgene                             name    _score  symbol    _id taxid
1       1017         cyclin-dependent kinase 2 371.45697    CDK2   1017  9606
2      12566         cyclin-dependent kinase 2 348.65656    Cdk2  12566 10090
3     362817         cyclin dependent kinase 2 263.61462    Cdk2 362817 10116
4      52004         CDK2-associated protein 2  20.85478 Cdk2ap2  52004 10090
5     143384 CDK2-associated, cullin domain 1  18.88728  CACUL1 143384  9606

$max_score
[1] 371.457

$took
[1] 676

$total
[1] 26
```

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

```
$hits
  entrezgene                                        name    _score symbol _id taxid
1        780 discoidin domain receptor tyrosine kinase 1 0.5057161   DDR1 780  9606

$max_score
[1] 0.5057161

$took
```

```
[1] 281

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

4 makeTranscriptDbFromMyGene

TranscriptDb is a container for storing transcript annotations. makeTranscriptDbFromMyGene allows the user to make a TranscriptDb object in the Genomic Features package from a mygene "exons" query.

```
> xli<-c('DDX26B',
+        'CCDC83',
+        'MAST3',
+        'RPL11',
+        'ZDHHC20',
+        'LUC7L3',
+        'SNORD49A',
+        'CTSH',
+        'ACOT8')
> txdb<-makeTranscriptDbFromMyGene(xli,
+           scopes="symbol", species="human")
> transcripts(txdb)
```

```
GRanges with 15 ranges and 2 metadata columns:
       seqnames                 ranges strand   |     tx_id      tx_name
          <Rle>              <IRanges>  <Rle>   | <integer>  <character>
   [1]        1   [24018268, 24022915]      +   |         5 NM_001199802
   [2]        1   [24018268, 24022915]      +   |         6    NM_000975
   [3]       11   [85566143, 85631063]      +   |         2 NM_001286159
   [4]       11   [85566143, 85631063]      +   |         3    NM_173556
   [5]       13   [21946709, 22033508]      -   |        10    NM_153251
   ...      ...                    ...    ... ...       ...          ...
  [11]       17 [ 48796925,  48830072]      +   |        11    NM_016424
  [12]       17 [ 48796925,  48830072]      +   |        12    NM_006107
  [13]       19 [ 18208602,  18262499]      +   |         4    NM_015016
  [14]       20 [ 44470359,  44486048]      -   |        15    NM_005469
  [15]        X [134654554, 134716460]      +   |         1    NM_182540
  ---
  seqlengths:
    X 11 19  1 13 17 15 20
   NA NA NA NA NA NA NA NA
```

makeTranscriptDbFromMyGene invokes either the query or queryMany method and passes the response to construct a `TranscriptDb` object. See `?TranscriptDb` 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
         query entrezgene         _id
   <character>  <numeric> <character>
1       DDX26B     203522      203522
2       CCDC83     220047      220047
3        MAST3      23031       23031
4        FLOT1      10211       10211
5        RPL11       6135        6135
6      ZDHHC20     253832      253832
7       LUC7L3      51747       51747
8     SNORD49A      26800       26800
9         CTSH       1512        1512
10       ACOT8      10005       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 3 columns
                                          ensembl.gene         _id       query
                                       <CharacterList> <character> <character>
1                                      ENSG00000165359      203522      DDX26B
2                                      ENSG00000150676      220047      CCDC83
3                                      ENSG00000099308       23031       MAST3
4  ENSG00000137312,ENSG00000206379,ENSG00000206480,...       10211       FLOT1
5                                      ENSG00000142676        6135       RPL11
6                                      ENSG00000180776      253832     ZDHHC20
7                                      ENSG00000108848       51747      LUC7L3
8                      ENSG00000277370,ENSG00000175061       26800    SNORD49A
```

```
9                                       ENSG00000103811       1512        CTSH
10                                      ENSG00000101473      10005       ACOT8
```

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

```
[1] "ENSG00000137312" "ENSG00000206379" "ENSG00000206480" "ENSG00000223654"
[5] "ENSG00000224740" "ENSG00000230143" "ENSG00000232280" "ENSG00000236271"
```

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>  <numeric> <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
        query entrezgene uniprot.Swiss-Prot           uniprot.TrEMBL         _id
  <character>  <numeric>        <character>          <CharacterList> <character>
1      DDX26B     203522             Q5JSJ4                               203522
2      CCDC83     220047             Q8IWF9                   H0YDV3      220047
3       MAST3      23031             O60307                   V9GYV0       23031
4       FLOT1      10211             O75955 A2AB09,A2AB10,A2AB11,...       10211
5       RPL11       6135             P62913     Q5VVC8,Q5VVC9,Q5VVD0        6135
6     Gm10494         NA                 NA                                   NA
7   1007_s_at  100616237                 NA                            100616237
8   1007_s_at        780             Q08345 A2ABL0,A2ABL2,A2ABM8,...         780
9    AK125780       2978             P43080                   A6PVH5        2978
   notfound
  <logical>
1        NA
2        NA
3        NA
4        NA
5        NA
6      TRUE
7        NA
8        NA
9        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
        query entrezgene uniprot.Swiss-Prot           uniprot.TrEMBL         _id
  <character>  <numeric>        <character>          <CharacterList> <character>
1      DDX26B     203522             Q5JSJ4                               203522
2      CCDC83     220047             Q8IWF9                   H0YDV3      220047
3       MAST3      23031             O60307                   V9GYV0       23031
4       FLOT1      10211             O75955 A2AB09,A2AB10,A2AB11,...       10211
5       RPL11       6135             P62913     Q5VVC8,Q5VVC9,Q5VVD0        6135
6     Gm10494         NA                 NA                                   NA
7   1007_s_at  100616237                 NA                            100616237
8   1007_s_at        780             Q08345 A2ABL0,A2ABL2,A2ABM8,...         780
9    AK125780       2978             P43080                   A6PVH5        2978
   notfound
  <logical>
1        NA
2        NA
3        NA
4        NA
5        NA
6      TRUE
7        NA
8        NA
9        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