

MyGene.info R Client

Adam Mark and Chunlei Wu

May 15, 2014

Contents

1	Overview	1
2	Gene Annotation Service	1
2.1	mg.getgene	1
2.2	mg.getgenes	2
3	Gene Query Service	3
3.1	mg.query	3
3.2	mg.querymany	4
4	Tutorial, ID mapping	6
4.1	Mapping gene symbols to Entrez gene ids	6
4.2	Mapping gene symbols to Ensembl gene ids	9
4.3	When an input has no matching gene	12
4.4	When input ids are not just symbols	14
4.5	When an input id has multiple matching genes	17
4.6	Can I convert a very large list of ids?	22
5	References	22

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 `mg.getgene`

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

```
> mg.getgene(1017, fields='name,symbol,taxid, entrezgene')
```

```
chr "name,symbol,taxid, entrezgene"
```

```
$`_id`
```

```
[1] "1017"
```

```
$symbol
```

```
[1] "CDK2"
```

```
$entrezgene
```

```
[1] 1017
```

```
$name
```

```
[1] "cyclin-dependent kinase 2"
```

```
$taxid
```

```
[1] 9606
```

2.2 `mg.getgenes`

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

```
> mg.getgenes(list(1017,1018,'ENSG00000148795'), species='human')
```

```
chr "human"
```

```
chr "symbol,name,taxid,entrezgene"
```

```
[[1]]
```

```
[[1]]$name
```

```
[1] "cyclin-dependent kinase 2"
```

```
[[1]]$symbol
```

```
[1] "CDK2"
```

```
[[1]]$taxid
```

```
[1] 9606
```

```
[[1]]$entrezgene
```

```
[1] 1017
```

```
[[1]]$query
```

```
[1] "1017"
```

```
[[1]]$`_id`  
[1] "1017"
```

```
[[2]]  
[[2]]$name  
[1] "cyclin-dependent kinase 3"
```

```
[[2]]$symbol  
[1] "CDK3"
```

```
[[2]]$taxid  
[1] 9606
```

```
[[2]]$entrezgene  
[1] 1018
```

```
[[2]]$query  
[1] "1018"
```

```
[[2]]$`_id`  
[1] "1018"
```

```
[[3]]  
[[3]]$name  
[1] "cytochrome P450, family 17, subfamily A, polypeptide 1"
```

```
[[3]]$symbol  
[1] "CYP17A1"
```

```
[[3]]$taxid  
[1] 9606
```

```
[[3]]$entrezgene  
[1] 1586
```

```
[[3]]$query  
[1] "ENSG00000148795"
```

```
[[3]]$`_id`  
[1] "1586"
```

3 Gene Query Service

3.1 `mg.query`

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

```
> mg.query('cdk2', size=5)
```

```
$hits
```

	entrezgene	name	_score	symbol	_id	taxid
1	1017	cyclin-dependent kinase 2	379.36148	CDK2	1017	9606
2	12566	cyclin-dependent kinase 2	355.49432	Cdk2	12566	10090
3	362817	cyclin dependent kinase 2	269.22430	Cdk2	362817	10116
4	52004	CDK2-associated protein 2	20.37188	Cdk2ap2	52004	10090
5	143384	CDK2-associated, cullin domain 1	19.41898	CACUL1	143384	9606

```
$total
```

```
[1] 26
```

```
$max_score
```

```
[1] 379.3615
```

```
$took
```

```
[1] 7
```

```
> mg.query('symbol:cdk2', species='human')
```

```
$hits
```

	entrezgene	name	_score	symbol	_id	taxid
1	1017	cyclin-dependent kinase 2	87.86334	CDK2	1017	9606

```
$total
```

```
[1] 1
```

```
$max_score
```

```
[1] 87.86334
```

```
$took
```

```
[1] 3
```

3.2 `mg.querymany`

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

```
> mg.querymany(list(1017, '695'), scopes='entrezgene', species='human')
```

Finished

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

```
[[1]]
```

```
[[1]]$name
```

```
[1] "cyclin-dependent kinase 2"
```

```
[[1]]$symbol
```

```
[1] "CDK2"
```

```
[[1]]$taxid
```

```
[1] 9606
```

```
[[1]]$entrezgene
```

```
[1] 1017
```

```
[[1]]$query
```

```
[1] "1017"
```

```
[[1]]$`_id`
```

```
[1] "1017"
```

```
[[2]]
```

```
[[2]]$name
```

```
[1] "Bruton agammaglobulinemia tyrosine kinase"
```

```
[[2]]$symbol
```

```
[1] "BTK"
```

```
[[2]]$taxid
```

```
[1] 9606
```

```
[[2]]$entrezgene
```

```
[1] 695
```

```
[[2]]$query
```

```
[1] "695"
```

```
[[2]]$`_id`
```

```
[1] "695"
```

```
> mg.querymany('1017,695', scopes='entrezgene', species=9606, returnall=TRUE)
```

Finished

\$out

\$out[[1]]

```
$out[[1]]$name
[1] "cyclin-dependent kinase 2"

$out[[1]]$symbol
[1] "CDK2"

$out[[1]]$taxid
[1] 9606

$out[[1]]$entrezgene
[1] 1017

$out[[1]]$query
[1] "1017"

$out[[1]]$`_id`
[1] "1017"

$out[[2]]
$out[[2]]$name
[1] "Bruton agammaglobulinemia tyrosine kinase"

$out[[2]]$symbol
[1] "BTK"

$out[[2]]$taxid
[1] 9606

$out[[2]]$entrezgene
[1] 695

$out[[2]]$query
[1] "695"

$out[[2]]$`_id`
[1] "695"

$dup
list()

$missing
list()
```

4 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.

4.1 Mapping gene symbols to Entrez gene ids

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

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

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

```
> mg.querymany(xli, scopes='symbol', fields='entrezgene', species='human')
```

Finished

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

```
[[1]]
```

```
[[1]]$query
```

```
[1] "DDX26B"
```

```
[[1]]$entrezgene
```

```
[1] 203522
```

```
[[1]]$`_id`
```

```
[1] "203522"
```

```
[[2]]
```

```
[[2]]$query
```

```
[1] "CCDC83"
```

```
[[2]]$entrezgene  
[1] 220047
```

```
[[2]]$`_id`  
[1] "220047"
```

```
[[3]]  
[[3]]$query  
[1] "MAST3"
```

```
[[3]]$entrezgene  
[1] 23031
```

```
[[3]]$`_id`  
[1] "23031"
```

```
[[4]]  
[[4]]$query  
[1] "FLOT1"
```

```
[[4]]$entrezgene  
[1] 10211
```

```
[[4]]$`_id`  
[1] "10211"
```

```
[[5]]  
[[5]]$query  
[1] "RPL11"
```

```
[[5]]$entrezgene  
[1] 6135
```

```
[[5]]$`_id`  
[1] "6135"
```

```
[[6]]  
[[6]]$query  
[1] "ZDHHC20"
```

```
[[6]]$entrezgene
```



```
[1] 253832
```

```
[[6]]$`_id`  
[1] "253832"
```

```
[[7]]  
[[7]]$query  
[1] "LUC7L3"
```

```
[[7]]$entrezgene  
[1] 51747
```

```
[[7]]$`_id`  
[1] "51747"
```

```
[[8]]  
[[8]]$query  
[1] "SNORD49A"
```

```
[[8]]$entrezgene  
[1] 26800
```

```
[[8]]$`_id`  
[1] "26800"
```

```
[[9]]  
[[9]]$query  
[1] "CTSH"
```

```
[[9]]$entrezgene  
[1] 1512
```

```
[[9]]$`_id`  
[1] "1512"
```

```
[[10]]  
[[10]]$query  
[1] "ACOT8"
```

```
[[10]]$entrezgene  
[1] 10005
```

```
[[10]]$`_id`  
[1] "10005"
```

4.2 Mapping gene symbols to Ensembl gene ids

Now if you want Ensembl gene ids back:

```
> mg.querymany(xli, scopes='symbol', fields='ensembl.gene', species='human')
```

Finished

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

```
[[1]]  
[[1]]$ensembl.gene  
[1] "ENSG00000165359" "ENSG00000268630"
```

```
[[1]]$`_id`  
[1] "203522"
```

```
[[1]]$query  
[1] "DDX26B"
```

```
[[2]]  
[[2]]$ensembl.gene  
[1] "ENSG00000150676"
```

```
[[2]]$`_id`  
[1] "220047"
```

```
[[2]]$query  
[1] "CCDC83"
```

```
[[3]]  
[[3]]$ensembl.gene  
[1] "ENSG00000099308"
```

```
[[3]]$`_id`  
[1] "23031"
```

```
[[3]]$query  
[1] "MAST3"
```

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

```
[[4]]$`_id`  
[1] "10211"
```

```
[[4]]$query  
[1] "FLOT1"
```

```
[[5]]  
[[5]]$ensembl.gene  
[1] "ENSG00000142676"
```

```
[[5]]$`_id`  
[1] "6135"
```

```
[[5]]$query  
[1] "RPL11"
```

```
[[6]]  
[[6]]$ensembl.gene  
[1] "ENSG00000180776"
```

```
[[6]]$`_id`  
[1] "253832"
```

```
[[6]]$query  
[1] "ZDHHC20"
```

```
[[7]]  
[[7]]$ensembl.gene  
[1] "ENSG00000108848"
```

```
[[7]]$`_id`  
[1] "51747"
```

```
[[7]]$query  
[1] "LUC7L3"
```

```
[[8]]
[[8]]$ensembl.gene
[1] "ENSG00000206956" "ENSG00000175061"
```

```
[[8]]$`_id`
[1] "26800"
```

```
[[8]]$query
[1] "SNORD49A"
```

```
[[9]]
[[9]]$ensembl.gene
[1] "ENSG00000103811"
```

```
[[9]]$`_id`
[1] "1512"
```

```
[[9]]$query
[1] "CTSH"
```

```
[[10]]
[[10]]$ensembl.gene
[1] "ENSG00000101473"
```

```
[[10]]$`_id`
[1] "10005"
```

```
[[10]]$query
[1] "ACOT8"
```

4.3 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<-list('DDX26B',
+ 'CCDC83',
+ 'MAST3',
+ 'FLOT1',
+ 'RPL11',
+ 'Gm10494')
> mg.querymany(xli, scopes='symbol', fields='entrezgene', species='human')
```

Finished

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

```
[[1]]
```

```
[[1]]$query
```

```
[1] "DDX26B"
```

```
[[1]]$entrezgene
```

```
[1] 203522
```

```
[[1]]$`_id`
```

```
[1] "203522"
```

```
[[2]]
```

```
[[2]]$query
```

```
[1] "CCDC83"
```

```
[[2]]$entrezgene
```

```
[1] 220047
```

```
[[2]]$`_id`
```

```
[1] "220047"
```

```
[[3]]
```

```
[[3]]$query
```

```
[1] "MAST3"
```

```
[[3]]$entrezgene
```

```
[1] 23031
```

```
[[3]]$`_id`
```

```
[1] "23031"
```

```
[[4]]
```

```
[[4]]$query
```

```
[1] "FLOT1"
```

```
[[4]]$entrezgene
```

```
[1] 10211
```

```
[[4]]$`_id`
```

```
[1] "10211"
```

```
[[5]]
[[5]]$query
[1] "RPL11"

[[5]]$entrezgene
[1] 6135

[[5]]$`_id`
[1] "6135"
```

```
[[6]]
[[6]]$query
[1] "Gm10494"

[[6]]$notfound
[1] TRUE
```

4.4 When input ids are not just symbols

```
> xli<-list('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:

```
> mg.querymany(xli, scopes='symbol,reporter,accession',
+             fields='entrezgene,uniprot', species='human')
```

Finished

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

```
[[1]]
[[1]]$query
[1] "DDX26B"

[[1]]$entrezgene
```

```
[1] 203522
```

```
[[1]]$uniprot
[[1]]$uniprot$`Swiss-Prot`
[1] "Q5JSJ4"
```

```
[[1]]$`_id`
[1] "203522"
```

```
[[2]]
[[2]]$query
[1] "CCDC83"
```

```
[[2]]$entrezgene
[1] 220047
```

```
[[2]]$uniprot
[[2]]$uniprot$`Swiss-Prot`
[1] "Q8IWF9"
```

```
[[2]]$`_id`
[1] "220047"
```

```
[[3]]
[[3]]$query
[1] "MAST3"
```

```
[[3]]$entrezgene
[1] 23031
```

```
[[3]]$uniprot
[[3]]$uniprot$`Swiss-Prot`
[1] "O60307"
```

```
[[3]]$uniprot$TrEMBL
[1] "V9GYV0"
```

```
[[3]]$`_id`
[1] "23031"
```

```
[[4]]
[[4]]$query
[1] "FLOT1"

[[4]]$entrezgene
[1] 10211

[[4]]$uniprot
[[4]]$uniprot$`Swiss-Prot`
[1] "O75955"

[[4]]$uniprot$TrEMBL
[1] "A2AB09" "A2AB10" "A2AB11" "A2AB12" "A2AB13" "A2ABJ5" "B4DVY7" "Q5ST80"

[[4]]$`_id`
[1] "10211"

[[5]]
[[5]]$query
[1] "RPL11"

[[5]]$entrezgene
[1] 6135

[[5]]$uniprot
[[5]]$uniprot$`Swiss-Prot`
[1] "P62913"

[[5]]$uniprot$TrEMBL
[1] "Q5VVD0"

[[5]]$`_id`
[1] "6135"

[[6]]
[[6]]$query
[1] "Gm10494"

[[6]]$notfound
[1] TRUE
```



```
[[7]]  
[[7]]$query  
[1] "1007_s_at"
```

```
[[7]]$entrezgene  
[1] 100616237
```

```
[[7]]$`_id`  
[1] "100616237"
```

```
[[8]]  
[[8]]$query  
[1] "1007_s_at"
```

```
[[8]]$entrezgene  
[1] 780
```

```
[[8]]$uniprot  
[[8]]$uniprot$`Swiss-Prot`  
[1] "Q08345"
```

```
[[8]]$uniprot$TrEMBL  
[1] "Q96T61" "Q96T62"
```

```
[[8]]$`_id`  
[1] "780"
```

```
[[9]]  
[[9]]$query  
[1] "AK125780"
```

```
[[9]]$entrezgene  
[1] 2978
```

```
[[9]]$uniprot  
[[9]]$uniprot$`Swiss-Prot`  
[1] "P43080"
```

```
[[9]]$uniprot$TrEMBL  
[1] "A6PVH5"
```

```
[[9]]$`_id`
[1] "2978"
```

4.5 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, together with the mapping output, from the returned result:

```
> mg.querymany(xli, scopes='symbol,reporter,accession',
+               fields='entrezgene,uniprot', species='human', returnall=TRUE)
```

```
Finished
```

```
$out
```

```
$out[[1]]
```

```
$out[[1]]$query
```

```
[1] "DDX26B"
```

```
$out[[1]]$entrezgene
```

```
[1] 203522
```

```
$out[[1]]$uniprot
```

```
$out[[1]]$uniprot$`Swiss-Prot`
```

```
[1] "Q5JSJ4"
```

```
$out[[1]]$`_id`
```

```
[1] "203522"
```

```
$out[[2]]
```

```
$out[[2]]$query
```

```
[1] "CCDC83"
```

```
$out[[2]]$entrezgene
```

```
[1] 220047
```

```
$out[[2]]$uniprot
```

```
$out[[2]]$uniprot$`Swiss-Prot`
```

```
[1] "Q8IWF9"
```

```
$out[[2]]$`_id`  
[1] "220047"
```

```
$out[[3]]  
$out[[3]]$query  
[1] "MAST3"
```

```
$out[[3]]$entrezgene  
[1] 23031
```

```
$out[[3]]$uniprot  
$out[[3]]$uniprot$`Swiss-Prot`  
[1] "060307"
```

```
$out[[3]]$uniprot$TrEMBL  
[1] "V9GYV0"
```

```
$out[[3]]$`_id`  
[1] "23031"
```

```
$out[[4]]  
$out[[4]]$query  
[1] "FLOT1"
```

```
$out[[4]]$entrezgene  
[1] 10211
```

```
$out[[4]]$uniprot  
$out[[4]]$uniprot$`Swiss-Prot`  
[1] "075955"
```

```
$out[[4]]$uniprot$TrEMBL  
[1] "A2AB09" "A2AB10" "A2AB11" "A2AB12" "A2AB13" "A2ABJ5" "B4DVY7" "Q5ST80"
```

```
$out[[4]]$`_id`  
[1] "10211"
```

```
$out[[5]]  
$out[[5]]$query
```

```
[1] "RPL11"

$out[[5]]$entrezgene
[1] 6135

$out[[5]]$uniprot
$out[[5]]$uniprot$`Swiss-Prot`
[1] "P62913"

$out[[5]]$uniprot$TrEMBL
[1] "Q5VVD0"

$out[[5]]$`_id`
[1] "6135"

$out[[6]]
$out[[6]]$query
[1] "Gm10494"

$out[[6]]$notfound
[1] TRUE

$out[[7]]
$out[[7]]$query
[1] "1007_s_at"

$out[[7]]$entrezgene
[1] 100616237

$out[[7]]$`_id`
[1] "100616237"

$out[[8]]
$out[[8]]$query
[1] "1007_s_at"

$out[[8]]$entrezgene
[1] 780

$out[[8]]$uniprot
$out[[8]]$uniprot$`Swiss-Prot`
```

```
[1] "Q08345"
```

```
$out[[8]]$uniprot$TrEMBL
```

```
[1] "Q96T61" "Q96T62"
```

```
$out[[8]]$`_id`
```

```
[1] "780"
```

```
$out[[9]]
```

```
$out[[9]]$query
```

```
[1] "AK125780"
```

```
$out[[9]]$entrezgene
```

```
[1] 2978
```

```
$out[[9]]$uniprot
```

```
$out[[9]]$uniprot$`Swiss-Prot`
```

```
[1] "P43080"
```

```
$out[[9]]$uniprot$TrEMBL
```

```
[1] "A6PVH5"
```

```
$out[[9]]$`_id`
```

```
[1] "2978"
```

```
$dup
```

```
$dup$`1007_s_at`
```

```
[1] 2
```

```
$dup$`1007_s_at`
```

```
[1] 2
```

```
$dup$`1007_s_at`
```

```
[1] 2
```

```
$dup$`1007_s_at`
```

```
[1] 2
```

```
$dup$`1007_s_at`
```

```
[1] 2
```

```
$dup$`1007_s_at`  
[1] 2
```

```
$missing  
$missing[[1]]  
[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).

4.6 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.

5 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