ENVISIONQuery

October 25, 2011

`ENVISIONQuery-package` *ENVISIONQuery.*

Description

ENVISIONQuery.

Details

Package:	ENVISIONQuery
Type:	Package
Version:	1.0.5
Date:	2011-03-26
License:	GPL-2
LazyLoad:	yes

Author(s)

Alex Lisovich, Roger Day

`ENVISIONQuery` *Launch a query against Envision, a web online query system...*

Description

Launch a query against Envision, a web online query system providing elaborated information for EnCORE services. Return the results into an R object.

Usage

```
ENVISIONQuery(ids=c("1553619_a_at", "1553497_at"), typeName="menu",
    serviceName="menu", toolName="menu", chunk=1000, details=TRUE,
    writeHTML=FALSE, testMe=FALSE,
    graphicMenu=getOption("menu.graphics"), formatIt=TRUE,
    options=list(), filter=list(), compact=TRUE, verbose=FALSE)
```

Arguments

ids	Depending on input type, IDs for desired objects as a character vector a character string containing the entire Enfin xml document or a name of an Enfin xml file.
typeName	Type of input ids. If 'menu' (default), a menu is constructed allowing to choose one of the available types.
serviceName	The Envision service name or a character vector of service names. In a latter case system compbines requests into pipeline using the output of current service request as an input for a next one. If 'menu' (default), the menu is contructed allowing to choose one of the available services.
toolName	The tool for a particular envision service. If 'menu' (default) and there is more than one tool available, a menu is contructed allowing to choose one of the available tools.
chunk	The number of IDs retrieved during a single query session. In case the number of IDs exceeds the chunk size, the whole ID set is retrieved during multiple query sessions overcoming the potential limitations on a single query session ID set size. If NA, a single session is used. Default is 1000.
details	If TRUE (default), a list of intermediate results is returned; otherwise, just the final query result. Note: not implemented at this time
writeHTML	If TRUE (default is FALSE), write the received intermediate HTML to files.
testMe	If TRUE (default is FALSE), assign default values and run.
graphicMenu	If TRUE (default is FALSE), use a GUI window for the pick menus.
formatIt	If TRUE (default), try to interpret the returned character table and structure the result. If false, the character string representing the entire enfinXML file returned by ENVISION. Note: formatting is implemented only for 'Probe2Uniprot' and 'ID conversion' services at this time. If false, the character string representing the entire enfinXML file returned by ENVISION.
options	The (optional)list each element of which represents the <name,value> pair used to apply the additional constraints when sending a query to a particular service (maximum number of pathways for Reactome service as an example)
filter	The (optional)list where the name of each element represents the formatted output data frame column on which filetering is to be performed ('organism.species', 'Microarray.platform' etc.) and the list element containing a character vector defining the set of values on which the merging (intersection) for a given column will be performed ('Homo sapiens' for 'organism.species', 'affy_hg_u133_plus_2' for 'Microarray.platform' etc.). The filtering is performed if the list is not empty and the formatIt=TRUE. Default is an empty list.
compact	If TRUE and the formatted output (formatIt = TRUE)is represented by a data frame, collapses the rows with duplicated match sets but different attributes into a single row with unique match set and an attribute list separated by comma for each attribute column. Default is TRUE.
verbose	If TRUE (default is FALSE), more debugging information is printed.

Value

The ENVISIONQueryResult structure or NULL if no results were found.

Note

For a pipeline of services, only unformatted output supported at this time.

Author(s)

Alex Lisovich, Roger Day

Examples

```
#### basic ENVISIONQuery request

#convert the Affy probeset IDs to UniProt IDs
res<-ENVISIONQuery(ids=c("1553619_a_at","1552276_a_at","202795_x_at"),
serviceName="ID Conversion",toolName="Affy2Uniprot",typeName="Affymetrix ID");
print(res);

#retrieve the pathways for given Uniprot ID(s)
res<-ENVISIONQuery(ids="P38398",serviceName="Reactome",typeName="Uniprot ID");
print(res[,-3]);

#retrieve protein-protein interactions
res<-ENVISIONQuery(ids="P38398",serviceName="Intact",typeName="Uniprot ID");
print(res[1:5,]);

#convert EnSembl IDs to Uniprot IDs
res<-ENVISIONQuery(ids=c("ENSP00000397145","ENSP00000269554"),serviceName="Picr",typeName
print(res);

#### ENVISIONQuery request using options and filters

#match Uniprot IDs to EnSembl and TrEMBL
options<-list("enfin-picr-search-database"=c("ENSEMBL_HUMAN","TREMBL"));
res<-ENVISIONQuery(ids="P38398",serviceName="Picr",options=options,typeName="Protein ID")
print(res);

#retrieve the pathways for given Uniprot ID(s)sorting them by coverage
#and calcultating the total protein count
options<-list("enfin-reactome-add-coverage"="true",
"enfin-reactome-sort-by-coverage"="true");
res<-ENVISIONQuery(ids="P38398",serviceName="Reactome",options=options,typeName="Uniprot
print(res[,-3]);

#convert the Affy probeset IDs to UniProt IDs restricting
#output by micro array type and organism species
filter<-list(Microarray.Platform="affy_hg_u133_plus_2",
organism.species="Homo sapiens");
res<-ENVISIONQuery(ids=c("1553619_a_at","1552276_a_at","202795_x_at"),filter=filter,
serviceName="ID Conversion",toolName="Affy2Uniprot",typeName="Affymetrix ID");
print(res);

#### pipeline of ENVISIONQuery requests. As of recent version,
#### output formatting for a pipelined request is not supported yet.
```

```
#Intact-Reactome cascaded request for UniProt ID(s).
IntactReactomeXML<-ENVISIONQuery(ids="P38398",
serviceName=c("Intact","Reactome"),typeName="Uniprot ID",verbose=TRUE);
#convert xml text into XMLDocument
#using XML package for further exploring
xmlDoc<-xmlTreeParse(IntactReactomeXML,useInternalNodes = TRUE, asText=TRUE);
class(xmlDoc);

#### interactive ENVISIONQuery requests
## Not run:
res<-ENVISIONQuery(ids=c("1553619_a_at","1552276_a_at","202795_x_at"),filter=filter,
serviceName="menu",toolName="menu",typeName="menu");
print(res);

## End(Not run)
```

`getInputTypes` *Get the list of available input types for a given tool.*

Description

Get the list of available input types for a given tool.

Usage

```
getInputTypes(tool, inputType="menu")
```

Arguments

`tool`	The tool handler
`inputType`	Either 'menu' or a valid input type. Default is 'menu'.

Details

The input types could be subdivided into the following three groups: 1. The list of IDs (types 'Uniprot ID', 'Affymetrix ID', 'Enquant ID' and 'Protein ID'). 2. The character string representing the enfinXML document ('Enfin XML') and 3. The name of the file containing the enfinXML document. The types belonging to the groups 2 and 3 can be used to construct a pipeline of queries, where the output of the given query can be used as an input for the next one

Value

If inputType is 'all', the character vector of all available input types. If inputType is valid input type, the inputType value returned, otherwise returns NULL.

Author(s)

Alex Lisovich, Roger Day

Examples

```
#check available input type for a given tool

service<-getService("Reactome");
tool<-getTool(service,"FindPathAdv");
getInputTypes(tool);
```

`getService` *Get the service handler using it's name...*

Description

Get the service handler using it's name

Usage

```
getService(serviceName="menu", graphicMenu=getOption("menu.graphics"))
```

Arguments

`serviceName`	The name of Envision service which handler should be retrieved. If equal to "menu" (default) menu is contructed allowing to choose one of the available services
`graphicMenu`	If TRUE (default is FALSE), use a GUI window for the pick menus.

Value

the Envision service handler.

Author(s)

Alex Lisovich, Roger Day

Examples

```
#get an ID Conversion service handle

service<-getService("ID Conversion");

## Not run:

#get a handle to service selected interactively

service<-getService("menu");

## End(Not run)
```

getServiceClient *Get the Web Service client finction handler for a given tool...*

Description

Get the Web Service client finction handler for a given tool

Usage

```
getServiceClient(tool)
```

Arguments

tool	The tool handler

Value

the function which implements client functionality for a particular data retrieval service

Author(s)

Alex Lisovich, Roger Day

Examples

```
#get Java Web Service client for 'FindPathAdv' tool of 'Reactome' service

service<-getService("Reactome");
getToolNames(service);
tool<-getTool(service,"FindPathAdv");
client<-getServiceClient(tool);
print(client);
```

getServiceNames *Get the names of available Envision services...*

Description

Get the names of available Envision services

Value

Character vector representing the list of of available Envision service names.

Author(s)

Alex Lisovich, Roger Day

Examples

```
#get the names of all available services
serviceNames<-getServiceNames();
print(serviceNames);
```

getServiceOptions *Get default service options.*

Description

Get default service options.

Usage

```
getServiceOptions(serviceName="menu",
    graphicMenu=getOption("menu.graphics"))
```

Arguments

serviceName	The name of Envision service which default options should be retrieved.
graphicMenu	If TRUE (default is FALSE), use a GUI window for the pick menus. If equal to "menu" (default) menu is contructed allowing to choose one of the available services

Value

The list where each item name and value corresponds to the particular option name and default value correspondingly.

Author(s)

Alex Lisovich, Roger Day

Examples

```
#get options for 'Reactome' service
options<-getServiceOptions("Reactome");
print(options);

## Not run:
#get options for a service selected interactively
getServiceOptions("menu");

## End(Not run)
```

getServices *Get available services...*

Description

Get available services

Value

The multi-level list of services and service attributes

Author(s)

Alex Lisovich, Roger Day

Examples

```
#get the handlers for all available services including utilities
services<-getServices();
names(services);
```

---

getTool *Get the tool handler of a given service using the tool name...*

Description

Get the tool handler of a given service using the tool name

Usage

```
getTool(service,toolName="menu",selection.title="Select Tool", graphicMenu=getOp
```

Arguments

service	Service handler
toolName	The name of Envision service tool handler to be retrieved. If equal to "menu" (default) and the number of tools is greater than 1 menu is contructed allowing to choose one of the available services.
selection.title	The selection list title. Default is 'Select Tool'.
graphicMenu	If TRUE (default is FALSE), use a GUI window for the pick menus.

Value

the Envision service tool handler.

Author(s)

Alex Lisovich, Roger Day

Examples

```
#get the tool handler for 'mapProteinsAdv' tool in 'Picr' service

service<-getService("Picr");
getToolNames(service);
tool<-getTool(service,"mapProteinsAdv");
```

`getToolNames` *Get the names of available tools for a given Envision service...*

Description

Get the names of available tools for a given Envision service

Usage

```
getToolNames(service)
```

Arguments

`service`	The Envision service handler

Value

Character vector representing the list of of available tool names for a given service.

Author(s)

Alex Lisovich, Roger Day

Examples

```
#get the list of available tools for a given service

service<-getService("ID Conversion");
toolNames<-getToolNames(service);
print(toolNames);
```

Index