SSN/76.2

Starlink Project
Starlink System Note 76.2

A.C. Davenhall

24 May 2001

CATREMOTE — a Tool for Querying Remote Catalogues

Abstract

catremote is a tool for querying remote astronomical catalogues, databases and archives via the Internet. It allows remote catalogues to be queried and the resulting table to be saved as a local file. It also provides a number of related auxiliary functions. catremote can be used interactively. However, it is more likely to be incorporated in a script or GUI. This document describes the use of catremote in sufficient detail to allow a programmer to use it in a script or GUI that he is developing.

Who Should Read this Document?

This document is aimed primarily at programmers who wish to incorporate catremote in a script or GUI that they are developing. It may also be useful to users who wish to simply run catremote interactively, particularly if they wish to make use of all its facilities. A simple introduction to catremote, which is more suitable for new users, is included in SUN/190.

Contents

List of Figures

Modes of operation

Mode	Description
`list`	list the databases currently available
`details`	show details of a named database
`query`	submit a query to a remote database and retrieve the results
`name`	resolve an object name into coordinates
`help`	list the modes available

Command-line arguments for the various modes

`catremote list`	*server-type*
`catremote details`	*db-name*
`catremote query`	*db-name* α δ *radius additional-condition*
`catremote name`	*db-name object-name*
`catremote help`	

The individual arguments are as follows.

server-type Type of server to be listed; one of: `all`, `catalog`, `archive`, `namesvr`, `imagesvr` or `local` (optional).

db-name Name of the database to be queried.

α Central Right Ascension of the query. The value should be for equinox J2000 and given in sexagesimal hours with a colon ('`:`') as the separator.

δ Central Declination of the query. The value should be for equinox J2000 and given in sexagesimal degrees with a colon ('`:`') as the separator. Southern Declinations are negative.

radius Radius of the query in minutes of arc.

additional-condition Any additional condition applied to the query (optional).

object-name The name of an astronmical objects whose coordinates are to be found. Give the name without embedded spaces and in either case (upper or lower).

Environment variables

Variable	Default	Description
`CATREM_URLREADER`		Program to submit query
`CATREM_CONFIG`		URL of configuration file
`CATREM_MAXOBJ`	1000	Maximum number of objects in results table
`CATREM_ECHOURL`	no	Echo URL sent to remote server?

Accessing this document

A hypertext version of this document is available. To access it on Starlink systems type:

```
% showme  ssn76
```

On non-Starlink systems access URL:

```
http://www.starlink.ac.uk/docs/ssn76.htx/ssn76.html
```

Obtaining assistance

`catremote` is part of the CURSA package for manipulating astronomical catalogues and tables. Reports of bugs in CURSA, including `catremote`, should always be sent to username `starlink@jiscmail.ac.uk`. However, you are welcome to contact me for advice and assistance; details of how to do so are given below.

Postal address:

A.C. Davenhall. Institute for Astronomy, Royal Observatory, Blackford Hill,
Edinburgh, EH9 3HJ, United Kingdom.

Electronic mail: `acd@roe.ac.uk`

Fax:

from within the United Kingdom:	0131-668-8416
from overseas:	+44-131-668-8416

Revision history

(1) 24th May 2001: Version 1. Original version (ACD).

1 Introduction

`catremote` is a tool for querying remote astronomical catalogues, databases and archives via the Internet. It allows remote catalogues to be queried and the resulting table to be saved as a local file. It also provides a number of related auxiliary functions.

`catremote` can be used interactively. However, it is more likely to be incorporated in a script or GUI. This document describes the use of `catremote` in sufficient detail to allow a programmer to use it in a script or GUI which he is developing. Additionally the document may be useful to users who simply wish to run `catremote` interactively, particularly if they wish to make use of all its facilities. *There is a simple introduction to the use of `catremote` in SUN/190[4] which may be more suitable for new users than the present document.*

`catremote` is part of the astronomical catalogue and table manipulation package CURSA (see SUN/190[4]). It is written in Perl and uses the 'ACL format' (discussed briefly below) to access remote catalogues and databases. A variety of remote catalogues and databases can be accessed using this format; in the first instance see SUN/190 for details of what is available.

2 The ACL Format

`catremote` uses the ACL format developed by Allan Brighton and colleagues at ESO to access remote catalogues and databases. The ACL format is fully documented in SSN/75[3]. The following brief description merely gives sufficient details to allow the operation of `catremote` to be understood.

The ACL format is implemented using the Hyper-Text Transfer Protocol (HTTP) developed as part of the World Wide Web. `catremote` querying a remote database is an example of a 'client-server architecture', with `catremote` acting as the client and the remote database, or more exactly the program which accesses it, as the server. In the ACL format a so-called 'configuration file' mediates the interaction between client and server. This configuration file comprises a list of one or more databases, giving details for each. Usually each 'database' will be a simple astronomical catalogue. However, other alternatives are possible: archives, name servers, *etc* (Table 4, below, lists all the possibilities). Consequently, in this document the generic term 'database' is used to denote each entry.

`catremote` accesses a given configuration file and the databases which this file contains are the ones that `catremote` currently knows about.

The configuration file lists various details for each database, such as: the URL to access it, the type of database it is (its so-called 'server type': catalogue, image server, name server *etc*), the type of queries supported, *etc*. Most of these details are not germane here. However, one item which is important is the so-called '`short_name`' or simply 'name' of the database. This quantity is used to identify the database, for example you would supply it in response to a prompt from `catremote`. It is a short character string (without embedded spaces) and conventionally it has the form:

database`@`*institution*

where *database* is an abbreviation for the database and *institution* a standardised abbreviation for the institution where the on-line version is located. By convention *institution* has three or four characters. For example, the usual name of the version of the USNO[1] PMM astrometric catalogue maintained by ESO is usno@eso.

3 System Requirements and Getting Started

catremote is available for all the versions of Unix supported by Starlink: Compaq Alpha/Tru64, Sun/Solaris and PC/Linux. It may well work on other versions of Unix, but is not supported on them.

catremote is a simple Perl script (see, for example, SUN/193[1]) and hence Perl must be available on your system before you can run it. It was developed using Perl 5. On Starlink systems catremote is located in file:

```
    /star/bin/cursa/catremote
```

On non-Starlink systems a release of catremote includes a copy of the script and this document. On such systems you may need to edit the first line of catremote to pick up Perl from wherever on your system it is located.

The purpose of catremote is to query remote catalogues via the Internet. The ACL format that it uses for this purpose itself uses the World Wide Web Hyper-Text Transfer Protocol (HTTP). Consequently, catremote will only run on computers connected to the Internet and configured to run Web browsers.

Though catremote assembles the queries and writes the resulting tables of selected objects, it runs a separate utility program to submit the query to the server and retrieve the results. A number of alternative utility programs can be used for this purpose, including the C program geturl which is supplied with catremote and is the default on Starlink systems. The various alternatives are described in Section 9. The Unix environment variable CATREM_URLREADER (below) specifies which utility program is to be used and must be set before catremote is invoked.

4 Environment Variables

catremote takes some input from Unix shell environment variables and these variables can be used to control its behaviour. Some of the variables are optional, but others are mandatory and must be set before catremote is invoked. The variables used are listed in Table 1 and described briefly below.

CATREM_URLREADER catremote uses a separate program to submit the URL constituting a query to the server and return the table of results. The options available are described in Section 9.

[1] http://www.nofs.navy.mil/

Variable	Default	Description
CATREM_URLREADER		Program to submit query
CATREM_CONFIG		URL of configuration file
CATREM_MAXOBJ	1000	Maximum number of objects in results table
CATREM_ECHOURL	no	Echo URL sent to remote server?

Table 1: Environment variables used by catremote. The variables for which no default is given are mandatory

CATREM_CONFIG specifies the configuration file to be used. It should be set to either the URL (for a remote file) or the local file name, including a directory specification (for a local file). Configuration files are described in SSN/75[3]. The default configuration file used by CURSA is at URL:

```
http://dev.starlink.ac.uk/~pwd/catremote/cursa.cfg
```

CATREM_MAXOBJ is the maximum number of objects which the returned table is allowed to contain.

CATREM_ECHOURL Controls whether the URL representing the query submitted to the remote server is also displayed to the user. The default is 'no'; to see the URL set CATREM_ECHOURL to 'yes'. Seeing the URL is potentially useful when debugging configuration files and servers but is not usually required for normal operation.

These environment variables are set up automatically when CURSA is started. However, if catremote is being used outside CURSA they need to be set up manually. Figure 1 shows an example script for this purpose. On Starlink systems it is available as file:

```
/star/share/cursa/cursacatremote-setup.csh
```

5 The Functionality of catremote

The purpose of catremote is to query remote astronomical databases. It has a number of separate functions to realise this purpose, and each function corresponds to a mode of the program. The mode to be used is specified when catremote is run. The various modes are listed in Table 2 and described briefly below. Running catremote to invoke the various modes is described in detail in Section 6, below.

list list all the databases which are currently available.

```
#! /bin/csh -f
#+
# catremote-setup.csh
#
# Example shell script to setup the environment variables used by
# catremote.  Note that this file should be sourced.
#
# Author:
#   A C Davenhall (Edinburgh)
# History:
#   23/5/01 (ACD): Original version.
#-

#  Define the utility program to be used to submit the query.  The options
#  are:
#    "/star/bin/cursa/geturl" -- geturl utility,
#    "lynx -source"           -- lynx command line browser,
#    "java  UrlReader"        -- Java utility.
#    "wget -q -O -"           -- wget utility (probably RedHat Linux only).

setenv  CATREM_URLREADER "/home/acd/starbase/cursa/catremote/geturl/geturl"

#  Specify the configuration file to be used.  The URL given here is the
#  the CURSA default.

setenv  CATREM_CONFIG  http://dev.starlink.ac.uk/~pwd/catremote/cursa.cfg

#  Define the maximum number of objects which may be included in the
#  returned table.

setenv  CATREM_MAXOBJ  200

#  Specify whether the URL constituting the query is echoed to the
#  command line.  The options re:
#    no   -  do not echo the URL (default),
#    yes  -  echo the URL.

setenv  CATREM_ECHOURL no

#  Set the Java CLASSPATH environment variable to pick up the URLreader
#  (note that CLASSPATH has to be set rather than setting the corresponding
#  command line option because the latter does not work on alphas).

setenv CLASSPATH /home/acd/starbase/cursa/catremote:/usr/lib/netscape/java/classes
```

Figure 1: Example shell script to set up `catremote` environment variables

`details` show details of a named database.

`query` submit a query to a named database and retrieve the results. The basic type of query supported is the 'cone search' or 'circular area search' which returns all the objects found in a given circular area of sky. This area is specified by its central Right Ascension and Declination and angular radius. The objects returned are formatted as a catalogue and written to a local file.

`name` submit a name of an astronomical object to a remote name-resolver database. If the name-resolver finds this name in its database then the Right Ascension and Declination of the object are returned and displayed.

`help` list the modes available.

Mode	Description
`list`	list the databases currently available
`details`	show details of a named database
`query`	submit a query to a remote database and retrieve the results
`name`	resolve an object name into coordinates
`help`	list the modes available

Table 2: The modes of `catremote`

6 Running catremote

`catremote` is invoked by simply typing:

```
% catremote
```

Arguments may be supplied on the command line or prompted for. Obviously, if `catremote` is being invoked from a script the arguments will usually be supplied on the command line.

The first command-line argument is the mode of operation. The permitted values are listed in Table 2. The mode can only be specified on the command line. If it is omitted then '`help`' mode is assumed.

The subsequent arguments required depend on the mode chosen and are summarised in Table 3. Command line arguments are identified by position. They may optionally be omitted, starting at the right. Omitted arguments will usually be prompted for. Exceptions are the *server-type* in `list` mode, any *additional-condition* in `query` mode and the mode itself (see below). For each mode the input required and output produced is described below.

`catremote list`	*server-type*
`catremote details`	*db-name*
`catremote query`	*db-name* α δ *radius additional-condition*
`catremote name`	*db-name object-name*
`catremote help`	

Table 3: Arguments for the various modes of `catremote`

6.1 `list` mode

Input `list` mode has a single optional argument, the *server-type*. The values permitted are listed in Table 4 (and SSN/75[3] gives more details of the individual types). `catremote` lists all the databases in the current configuration file which match the given *server-type*. If the argument is omitted then '`all`' is assumed (that is, all the databases in the configuration file are listed).

Server type	Description
`all`	all types of server
`catalog`	simple catalogue
`archive`	archive
`namesvr`	name server
`imagesvr`	image server
`local`	local file
`directory`	link to another configuration file

Table 4: Types of server

Output All the databases in the configuration file which match the specified server type are listed, one per line. Typical output might look something like:

```
usno@eso  catalog  USNO at ESO
gsc@lei  catalog  Guide Star Catalog at LEDAS
simbad_ns@eso  namesvr  SIMBAD Names
```

For each database, the first item is its name, the second its server type and the rest of the line gives a short description.

6.2 `details` mode

Input `details` mode has a single input argument, *db-name*; the name of the database for which the details are to be shown (see Section 2). This database must be included in the current configuration file.

Output The following details are shown for the specified database, listed one per line:

`short_name:`	name of the database
`serv_type:`	server type of the database (see Table 4)
`long_name:`	a short description of the database
`url:`	URL of the database server
`search_cols:`	columns on which range searches are supported
`help:`	URL of help page for the database

See SSN/75[3] for a description of each item.

6.3 `query` mode

Input `query` mode has the arguments described below. Any arguments which are omitted will be prompted for, with the exception of *additional-condition*. Thus, if an *additional-condition* is specified then all the arguments must be included on the command line.

db-name Name of the database to be queried (see Section 2).

α Central Right Ascension of the query. The value should be for equinox J2000 and given in sexagesimal hours with a colon (‘`:`’) as the separator.

δ Central Declination of the query. The value should be for equinox J2000 and given in sexagesimal degrees with a colon (‘`:`’) as the separator. Southern Declinations are negative.

radius Radius of the query in minutes of arc.

additional-condition Any additional condition applied to the query. Databases vary in which, if any, additional queries they support. Three forms of *additional-condition* are accepted. The first is:

> *column-name=minimum-value,maximum-value*

and objects will only be selected if their value for column *column-name* lies between *minimum-value* and *maximum-value*. The second is:

> *minimum-magnitude,maximum-magnitude*

here the range is assumed to be a magnitude and no column name is specified (remember that magnitudes increase the ‘wrong way round’ so that *minimum-magnitude* corresponds to the brightest object). The third is:

> *maximum-magnitude*

which is again assumed to be a magnitude and only objects brighter than *maximum-magnitude* are selected. For completeness, the relationship between the forms of *additional-condition* and the ‘query tokens’ for the database specified in the configuration file (as described in SSN/75[3]) is that *additional-condition* replaces the tokens as follows:

	Form	replaces token
first:	*column-name=minimum-value,maximum-value*	`%cond`
second:	*minimum-magnitude,maximum-magnitude*	`%m1`, `%m2`
third:	*maximum-magnitude*	`%m`

Output The objects selected are written as a catalogue in the Tab-Separated Table (TST) format in the current directory. The TST format is described in SSN/75[3]. The name of the catalogue file is derived automatically from the name of the database and the central Right Ascension and Declination. `catremote` displays a line showing the name of the file created, for example:

```
!(Info.) Catalogue usno_eso_101500_303000.tab written successfully.
```

If the catalogue contained no objects which satisfied the query `catremote` will report:

```
! Failure: no objects found in the region specified.
```

6.4 `name` mode

Input `name` mode has the following two arguments. If they are omitted then they will be prompted for.

db-name The name of the name resolver database which is to be queried (see Section 2). The usual choice is `simbad_ns@eso`, the SIMBAD name resolver provided by ESO using the SIMBAD[2] integrated database maintained by the Centre de Données astronomiques de Strasbourg (CDS).

object-name The name of an astronomical object which is to be resolved. It should be entered without embedded spaces. The case of letters (upper or lower) is not usually significant. That is, case is not significant for `simbad_ns@eso` and probably will not be significant for other name resolvers.

Output If the name resolver resolves the given name, that is successfully looks it up and finds coordinates for it, then they are displayed, for example:

```
Right Ascension: +10:47:50
Declination: +12:34:57
```

The Right Ascension is in sexagesimal hours, the Declination in sexagesimal degrees and both are for equinox J2000. If the name could not be resolved then `catremote` reports:

```
! Failure: unable to resolve object name.
```

6.5 `help` mode

Input No arguments are required.

Output All the various modes are listed, with a one-line summary of each.

[2] http://simbad.u-strasbg.fr/Simbad

7 Interactive Examples

This section gives some examples of using `catremote` interactively. All the arguments are given on the command line. However, if they are omitted then arguments other than the first (the mode) will usually be prompted for.

(1) List the various modes in which `catremote` may be used. Type either of:

```
% catremote
% catremote help
```

(2) List all the databases in the current configuration file:

```
% catremote list
```

(3) List all the name servers (that is, databases of server type ‘`namesvr`’) in the current configuration file:

```
% catremote list namesvr
```

(4) Show details of the USNO PMM astrometric catalogue:

```
% catremote details usno@eso
```

(5) Find all the objects in the USNO PMM which lie within ten minutes of arc of Right Ascension $12^{\mathrm{h}}\,15^{\mathrm{m}}\,00\overset{\mathrm{s}}{.}0$ and Declination $30^{\circ}\,30'\,00\overset{\prime\prime}{.}0$ (J2000):

```
% catremote query usno@eso 12:15:00 30:30:00 10
```

The objects selected will be saved as a catalogue called `usno_eso_121500_303000.tab` created in your current directory. This catalogue will be written in the Tab-Separated Table (TST) format.

(6) Find all the objects in the USNO PMM which lie within ten minutes of arc of Right Ascension $12^{\mathrm{h}}\,15^{\mathrm{m}}\,00\overset{\mathrm{s}}{.}0$ and Declination $30^{\circ}\,30'\,00\overset{\prime\prime}{.}0$ (J2000) which also lie in the magnitude range 14 to 16:

```
% catremote query usno@eso 12:15:00 30:30:00 10 14,16
```

(7) Find the equatorial coordinates of the galaxy NGC 3379:

```
% catremote name simbad_ns@eso ngc3379
```

The coordinates returned are for equinox J2000.

8 Scripting Examples

Some complete examples which illustrate the use of `catremote` in scripts are available. They are written in the tcl scripting language[3] and on Starlink systems are in directory:

```
/star/share/cursa
```

The scripts provided are:

`listavaildb.tcl` list the databases currently available,

`querycat.tcl` query a remote catalogue,

`resolvename.tcl` find the coordinates of a named object.

On Starlink systems the scripts should simply run without requiring any modifications. On non-Starlink systems you may need to edit them to change the specified location of both the tcl interpreter and the `catremote` utility. In all cases the former is the first line of the script and the latter the first executable line after the introductory comments. Sufficient comments should be included to document the use of `catremote` in each script.

The utility `findcoords` (see SUN/240[5]) is a simple wrap-around for the name-resolver function of `catremote`. It can serve as an example of invoking `catremote` from a Perl script. On Starlink systems the source is available in file:

```
/star/bin/findcoords
```

9 Remote Access Utility

`catremote` runs a separate utility program to submit the URL representing the query of a remote catalogue and retrieve the results. The utility to be used is not hard-wired into `catremote`, but rather is specified using environment variable `CATREM_URLREADER` (see Section 4). Several options are available:

- `geturl`, a C program supplied with `catremote`,
- `UrlReader`, a Java program also supplied with `catremote`,
- the `lynx`[4] command-line browser,
- the `wget` URL-retrieval tool.

[3] Tcl is described by its author, John Ousterhout, in his *Tcl and the Tk Toolkit*[6]; see also SUN/200[2].

[4] http://lynx.browser.org/

geturl is the default on Starlink systems. It is reasonably fast, as are lynx and and wget, though UrlReader seems somewhat slower. Brief details of using the various options follow.

geturl specify:

```
% setenv CATREM_URLREADER "/star/bin/cursa/geturl"
```

UrlReader specify:

```
% setenv CATREM_URLREADER  "java  UrlReader"
```

It is also necessary to set the java environment variable CLASSPATH so that UrlReader is picked up in addition to the standard Java classes. For example, I might set:

```
% setenv CLASSPATH /star/bin/cursa:/usr/lib/netscape/java/classes
```

Note that it is necessary to specify the location of UrlReader using CLASSPATH rather than the corresponding Java command-line option because the latter appears not to work on Compaq Alpha/Tru64.

lynx specify:

```
% setenv  CATREM_URLREADER  "lynx -source"
```

wget specify:

```
% setenv  CATREM_URLREADER  "wget -q -O -"
```

wget is distributed as part of RedHat Linux systems and will probably only be available as part of such systems. On these systems an on-line manual can be accessed by typing:

```
% info  wget
```

Acknowledgements

I am grateful to Allan Brighton, Martin Bly, Peter Draper, Horst Meyerdierks and Mike Read for either advice or comments on the draft version of the document. Any mistakes, of course, are my own.

References

[1] M.J. Bly, 16 June 1999, SUN/193.4: *PERL — Practical Extraction and Report Language*, Starlink. 3

[2] M.J. Bly, 13 October 2000, SUN/200.3: *TCLSYS — TCL, TK and EXPECT utilities*, Starlink. 3

[3] A.C. Davenhall, 26 July 2000, SSN/75.1: *Writing Catalogue and Image Servers for GAIA and CURSA*, Starlink. 2, 4, 6.1, 6.2, 6.3

[4] A.C. Davenhall, 14 May 2001, SUN/190.9: *CURSA — Catalogue and Table Manipulation Applications*, Starlink. 1

[5] A.C. Davenhall, 29 May 2001, SUN/240.1: *FINDCOORDS — Finding the Coordinates of a Named Object*, Starlink. 8

[6] J.K.Ousterhout, 1994, *Tcl and the Tk Toolkit* (Addison-Wesley: Reading, Massachusetts). 3

A Detailed Description of Applications

CATREMOTE
A simple script to query remote catalogues

Description:

catremote is a tool for querying remote astronomical catalogues, databases and archives via the Internet. It allows remote catalogues to be queried and the resulting table saved as a local file written in the Tab-Separated Table (TST) format. It also provides a number of related auxiliary functions.

catremote has several different modes of usage, each providing a different function. The modes are:

list - list the catalogues currently available,

details - show details of a named catalogue,

query - submit a query to a remote catalogue and retrieve the results,

name - resolve an object name into coordinates,

help - list the modes available.

There is an introduction to using catremote in SUN/190 and it is comprehensively documented in SSN/76.

Usage:

```
Arguments for catremote can be specified on the command line.  If arguments
other than the first are omitted then they will usually be prompted for.
The first argument is the mode of operation and its value determines the
other arguments which are required.  The arguments for the various modes
are:

catremote list server-type

catremote details db-name

catremote query db-name alpha delta radius additional-condition

catremote name db-name object-name

catremote help

The individual arguments are described in the ‘Arguments’ section.  If the
mode is omitted then ’help’ mode is assumed.

In addition to the command-line arguments, catremote takes some input from Unix
shell environment variables and these variables can be used to control its behaviour.
```

Examples:

```
catremote
```

```
catremote help
```

List the various modes in which catremote may be used.

```
catremote list
```

List all the catalogues and databases in the current configuration file.

```
catremote list namesvr
```

List all the name servers (that is, databases of server type 'namesvr') in the current configuration file.

```
catremote details usno@eso
```

Show details of the USNO PMM astrometric catalogue (whose name is 'usno@eso').

```
catremote query usno@eso 12:15:00 30:30:00 10
```

Find all the objects in the USNO PMM which lie within ten minutes of arc of Right Ascension 12:15:00.0 (sexagesimal hours) and Declination 30:30:00.0 (sexagesimal degrees, both J2000). The objects selected will be saved as a catalogue called usno_eso_121500_303000.tab created in your current directory. This catalogue will be written in the Tab-Separated Table (TST) format.

```
catremote query usno@eso 12:15:00 30:30:00 10 14,16
```

Find all the objects in the USNO PMM which lie within ten minutes of arc of Right Ascension 12:15:00.0 (sexagesimal hours) and Declination 30:30:00.0 (sexagesimal degrees, both J2000) which also lie in the magnitude range 14 to 16.

```
catremote name simbad_ns@eso ngc3379
```

Find the equatorial coordinates of the galaxy NGC 3379. The coordinates returned are for equinox J2000.

Environment Variables :

CATREM_URLREADER (read) catremote uses a separate program to submit the URL constituting a query to the server and return the table of results. This environment variable specifies the program to be used. See SSN/76 for further details. (Mandatory.)

CATREM_CONFIG (read) This environment variable specifies the configuration file to be used. It should be set to either the URL (for a remote file) or the local file name, including a directory specification (for a local file). Configuration files mediate the interaction between catremote and the remote catalogue; see SSN/76 for further details. (Mandatory.)

CATREM_MAXOBJ (read) The maximum number of objects which the returned table is allowed to contain.

CATREM_ECHOURL (read) This environment controls whether the URL representing the query submitted to the remote catalogue is also displayed to the user. The default is `'no'`; to see the URL set CATREM_ECHOURL to `'yes'`. Seeing the URL is potentially useful when debugging configuration files and remote catalogue servers but is not usually required for normal operation.

GETURL
Retrieve a specified URL and write it to standard output

Description:

Retrieve the contents of a specified URL and write them to standard output.

Usage:

```
geturl url-to-be-retrieved [show-HTTP-header]
```

Parameters:

url-to-be-retrieved

The URL whose contents are to be written to standard output.

show-HTTP-header

If any value is given for this optional argument then the HTTP header at the start of the requested page (which is usually hidden) is echoed to standard output.

Examples:

```
geturl http://www.roe.ac.uk/acdwww/cursa/home.html
```

Retrieve the contents of URL http://www.roe.ac.uk/acdwww/cursa/home.html.

```
geturl http://www.roe.ac.uk/acdwww/cursa/home.html head
```

Retrieve the contents of URL http://www.roe.ac.uk/acdwww/cursa/home.html and show the header (which is usually hidden) at the start of the page.