00 魔法命令

2023 年 10 月 17 日

[1]:
```
%lsmagic
```

[1]:
```
Available line magics:
%alias  %alias_magic  %autoawait  %autocall  %automagic  %autosave  %bookmark
%cd  %clear  %cls  %code_wrap  %colors  %conda  %config  %connect_info  %copy
%ddir  %debug  %dhist  %dirs  %doctest_mode  %echo  %ed  %edit  %env  %gui
%hist  %history  %killbgscripts  %ldir  %less  %load  %load_ext  %loadpy
%logoff  %logon  %logstart  %logstate  %logstop  %ls  %lsmagic  %macro  %magic
%matplotlib  %mkdir  %more  %notebook  %page  %pastebin  %pdb  %pdef  %pdoc
%pfile  %pinfo  %pinfo2  %pip  %popd  %pprint  %precision  %prun  %psearch
%psource  %pushd  %pwd  %pycat  %pylab  %qtconsole  %quickref  %recall  %rehashx
%reload_ext  %ren  %rep  %rerun  %reset  %reset_selective  %rmdir  %run  %save
%sc  %set_env  %store  %sx  %system  %tb  %time  %timeit  %unalias  %unload_ext
%who  %who_ls  %whos  %xdel  %xmode

Available cell magics:
%%!  %%HTML  %%SVG  %%bash  %%capture  %%cmd  %%code_wrap  %%debug  %%file
%%html  %%javascript  %%js  %%latex  %%markdown  %%perl  %%prun  %%pypy
%%python  %%python2  %%python3  %%ruby  %%script  %%sh  %%svg  %%sx  %%system
%%time  %%timeit  %%writefile

Automagic is ON, % prefix IS NOT needed for line magics.
```

[2]:
```
%quickref
```

```
IPython -- An enhanced Interactive Python - Quick Reference Card
================================================================

obj?, obj??      : Get help, or more help for object (also works as
```

```
                  ?obj, ??obj).
?foo.*abc*       : List names in 'foo' containing 'abc' in them.
%magic           : Information about IPython's 'magic' % functions.

Magic functions are prefixed by % or %%, and typically take their arguments
without parentheses, quotes or even commas for convenience.  Line magics take a
single % and cell magics are prefixed with two %%.

Example magic function calls:

%alias d ls -F   : 'd' is now an alias for 'ls -F'
alias d ls -F    : Works if 'alias' not a python name
alist = %alias   : Get list of aliases to 'alist'
cd /usr/share    : Obvious. cd -<tab> to choose from visited dirs.
%cd??            : See help AND source for magic %cd
%timeit x=10     : time the 'x=10' statement with high precision.
%%timeit x=2**100
x**100           : time 'x**100' with a setup of 'x=2**100'; setup code is not
                  counted.  This is an example of a cell magic.

System commands:

!cp a.txt b/     : System command escape, calls os.system()
cp a.txt b/      : after %rehashx, most system commands work without !
cp ${f}.txt $bar : Variable expansion in magics and system commands
files = !ls /usr : Capture system command output
files.s, files.l, files.n: "a b c", ['a','b','c'], 'a\nb\nc'

History:

_i, _ii, _iii    : Previous, next previous, next next previous input
_i4, _ih[2:5]    : Input history line 4, lines 2-4
exec(_i81)       : Execute input history line #81 again
%rep 81          : Edit input history line #81
_, __, ___       : previous, next previous, next next previous output
_dh              : Directory history
_oh              : Output history
```

```
%hist            : Command history of current session.
%hist -g foo     : Search command history of (almost) all sessions for 'foo'.
%hist -g         : Command history of (almost) all sessions.
%hist 1/2-8      : Command history containing lines 2-8 of session 1.
%hist 1/ ~2/     : Command history of session 1 and 2 sessions before current.
%hist ~8/1-~6/5  : Command history from line 1 of 8 sessions ago to
                   line 5 of 6 sessions ago.
%edit 0/         : Open editor to execute code with history of current session.

Autocall:

f 1,2            : f(1,2)  # Off by default, enable with %autocall magic.
/f 1,2           : f(1,2) (forced autoparen)
,f 1 2           : f("1","2")
;f 1 2           : f("1 2")

Remember: TAB completion works in many contexts, not just file names
or python names.

The following magic functions are currently available:

%alias:
    Define an alias for a system command.
%alias_magic:
    ::
%autoawait:

%autocall:
    Make functions callable without having to type parentheses.
%automagic:
    Make magic functions callable without having to type the initial %.
%autosave:
    Set the autosave interval in the notebook (in seconds).
%bookmark:
    Manage IPython's bookmark system.
%cd:
    Change the current working directory.
```

```
%clear:
    Clear the terminal.
%cls:
    Clear the terminal.
%code_wrap:
    ::
%colors:
    Switch color scheme for prompts, info system and exception handlers.
%conda:
    Run the conda package manager within the current kernel.
%config:
    configure IPython
%connect_info:
    Print information for connecting other clients to this kernel
%copy:
    Alias for `!copy`
%ddir:
    Alias for `!dir /ad /on`
%debug:
    ::
%dhist:
    Print your history of visited directories.
%dirs:
    Return the current directory stack.
%doctest_mode:
    Toggle doctest mode on and off.
%echo:
    Alias for `!echo`
%ed:
    Alias for `%edit`.
%edit:
    Bring up an editor and execute the resulting code.
%env:
    Get, set, or list environment variables.
%gui:
    Enable or disable IPython GUI event loop integration.
%hist:
```

```
    Alias for `%history`.
%history:
    ::
%killbgscripts:
    Kill all BG processes started by %%script and its family.
%ldir:
    Alias for `!dir /ad /on`
%less:
    Show a file through the pager.
%load:
    Load code into the current frontend.
%load_ext:
    Load an IPython extension by its module name.
%loadpy:
    Alias of `%load`
%logoff:
    Temporarily stop logging.
%logon:
    Restart logging.
%logstart:
    Start logging anywhere in a session.
%logstate:
    Print the status of the logging system.
%logstop:
    Fully stop logging and close log file.
%ls:
    Alias for `!dir /on`
%lsmagic:
    List currently available magic functions.
%macro:
    Define a macro for future re-execution. It accepts ranges of history,
%magic:
    Print information about the magic function system.
%matplotlib:
    ::
%mkdir:
    Alias for `!mkdir`
```

```
%more:
    Show a file through the pager.
%notebook:
    ::
%page:
    Pretty print the object and display it through a pager.
%pastebin:
    Upload code to dpaste.com, returning the URL.
%pdb:
    Control the automatic calling of the pdb interactive debugger.
%pdef:
    Print the call signature for any callable object.
%pdoc:
    Print the docstring for an object.
%pfile:
    Print (or run through pager) the file where an object is defined.
%pinfo:
    Provide detailed information about an object.
%pinfo2:
    Provide extra detailed information about an object.
%pip:
    Run the pip package manager within the current kernel.
%popd:
    Change to directory popped off the top of the stack.
%pprint:
    Toggle pretty printing on/off.
%precision:
    Set floating point precision for pretty printing.
%prun:
    Run a statement through the python code profiler.
%psearch:
    Search for object in namespaces by wildcard.
%psource:
    Print (or run through pager) the source code for an object.
%pushd:
    Place the current dir on stack and change directory.
%pwd:
```

```
    Return the current working directory path.
%pycat:
    Show a syntax-highlighted file through a pager.
%pylab:
    ::
%qtconsole:
    Open a qtconsole connected to this kernel.
%quickref:
    Show a quick reference sheet
%recall:
    Repeat a command, or get command to input line for editing.
%rehashx:
    Update the alias table with all executable files in $PATH.
%reload_ext:
    Reload an IPython extension by its module name.
%ren:
    Alias for `!ren`
%rep:
    Alias for `%recall`.
%rerun:
    Re-run previous input
%reset:
    Resets the namespace by removing all names defined by the user, if
%reset_selective:
    Resets the namespace by removing names defined by the user.
%rmdir:
    Alias for `!rmdir`
%run:
    Run the named file inside IPython as a program.
%save:
    Save a set of lines or a macro to a given filename.
%sc:
    Shell capture - run shell command and capture output (DEPRECATED use !).
%set_env:
    Set environment variables.  Assumptions are that either "val" is a
%store:
    Lightweight persistence for python variables.
```

```
%sx:
    Shell execute - run shell command and capture output (!! is short-hand).
%system:
    Shell execute - run shell command and capture output (!! is short-hand).
%tb:
    Print the last traceback.
%time:
    Time execution of a Python statement or expression.
%timeit:
    Time execution of a Python statement or expression
%unalias:
    Remove an alias
%unload_ext:
    Unload an IPython extension by its module name.
%who:
    Print all interactive variables, with some minimal formatting.
%who_ls:
    Return a sorted list of all interactive variables.
%whos:
    Like %who, but gives some extra information about each variable.
%xdel:
    Delete a variable, trying to clear it from anywhere that
%xmode:
    Switch modes for the exception handlers.
%%!:
    Shell execute - run shell command and capture output (!! is short-hand).
%%HTML:
    Alias for `%%html`.
%%SVG:
    Alias for `%%svg`.
%%bash:
    %%bash script magic
%%capture:
    ::
%%cmd:
    %%cmd script magic
%%code_wrap:
```

```
    ::
%%debug:
    ::
%%file:
    Alias for `%%writefile`.
%%html:
    ::
%%javascript:
    Run the cell block of Javascript code
%%js:
    Run the cell block of Javascript code
%%latex:
    Render the cell as a block of LaTeX
%%markdown:
    Render the cell as Markdown text block
%%perl:
    %%perl script magic
%%prun:
    Run a statement through the python code profiler.
%%pypy:
    %%pypy script magic
%%python:
    %%python script magic
%%python2:
    %%python2 script magic
%%python3:
    %%python3 script magic
%%ruby:
    %%ruby script magic
%%script:
    ::
%%sh:
    %%sh script magic
%%svg:
    Render the cell as an SVG literal
%%sx:
    Shell execute - run shell command and capture output (!! is short-hand).
```

```
%%system:
    Shell execute - run shell command and capture output (!! is short-hand).
%%time:
    Time execution of a Python statement or expression.
%%timeit:
    Time execution of a Python statement or expression
%%writefile:
    ::
```

[3]:
```
%matplotlib?
```

```
Docstring:
::

  %matplotlib [-l] [gui]

Set up matplotlib to work interactively.

This function lets you activate matplotlib interactive support
at any point during an IPython session. It does not import anything
into the interactive namespace.

If you are using the inline matplotlib backend in the IPython Notebook
you can set which figure formats are enabled using the following::

    In [1]: from matplotlib_inline.backend_inline import set_matplotlib_formats

    In [2]: set_matplotlib_formats('pdf', 'svg')

The default for inline figures sets `bbox_inches` to 'tight'. This can
cause discrepancies between the displayed image and the identical
image created using `savefig`. This behavior can be disabled using the
`%config` magic::

    In [3]: %config InlineBackend.print_figure_kwargs = {'bbox_inches':None}

In addition, see the docstrings of
```

```
`matplotlib_inline.backend_inline.set_matplotlib_formats` and
`matplotlib_inline.backend_inline.set_matplotlib_close` for more information on
changing additional behaviors of the inline backend.

Examples
--------
To enable the inline backend for usage with the IPython Notebook::

    In [1]: %matplotlib inline

In this case, where the matplotlib default is TkAgg::

    In [2]: %matplotlib
    Using matplotlib backend: TkAgg

But you can explicitly request a different GUI backend::

    In [3]: %matplotlib qt

You can list the available backends using the -l/--list option::

   In [4]: %matplotlib --list
   Available matplotlib backends: ['osx', 'qt4', 'qt5', 'gtk3', 'gtk4',␣
 ↪'notebook', 'wx', 'qt', 'nbagg',
   'gtk', 'tk', 'inline']

positional arguments:
  gui         Name of the matplotlib backend to use ('agg', 'gtk', 'gtk3',␣
 ↪'gtk4', 'inline', 'ipympl', 'nbagg',
              'notebook', 'osx', 'pdf', 'ps', 'qt', 'qt4', 'qt5', 'qt6', 'svg',␣
 ↪'tk', 'webagg', 'widget', 'wx'). If
              given, the corresponding matplotlib backend is used, otherwise it␣
 ↪will be matplotlib's default (which
              you can set in your matplotlib config file).

options:
```

```
  -l, --list  Show available matplotlib backends
File:      d:\py_groundwater\lib\site-packages\ipython\core\magics\pylab.py
```

[4]:
```
%matplotlib --list
```

```
Available matplotlib backends: ['tk', 'gtk', 'gtk3', 'gtk4', 'wx', 'qt4', 'qt5',
'qt6', 'qt', 'osx', 'nbagg', 'webagg', 'notebook', 'agg', 'svg', 'pdf', 'ps',
'inline', 'ipympl', 'widget']
```

[ ]:
```

```