Python in the IDE
Requires reticulate plus RStudio v1.2 or higher.

Syntax highlighting for Python scripts and chunks

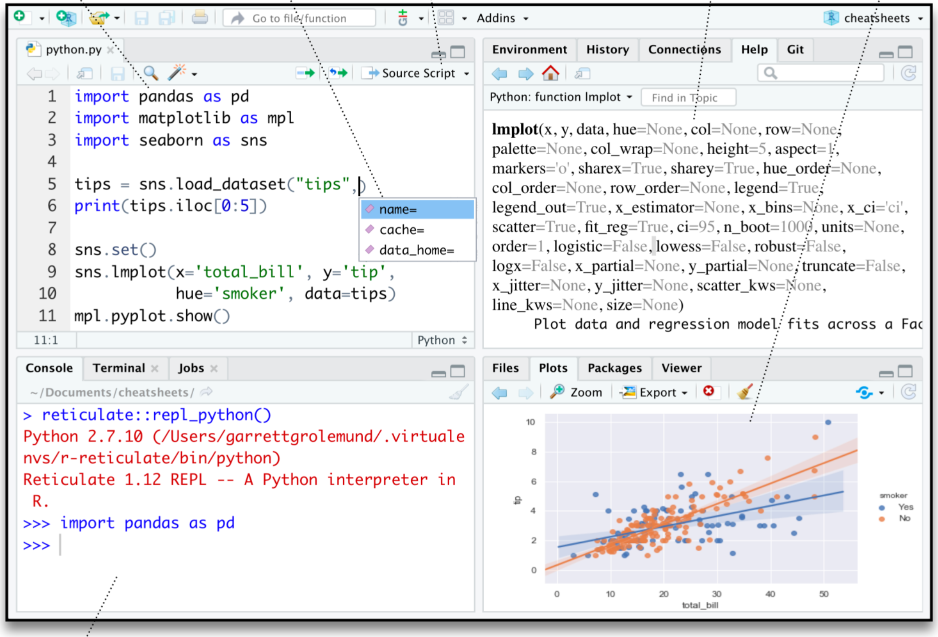
Tab completion for Python functions and objects (and Python modules imported in R scripts)

Source Python scripts.

Execute Python code line by line with **Cmd + Enter** (**Ctrl + Enter**)

Press **F1** over a Python symbol to display the help topic for that symbol.

matplotlib plots display in plots pane.



```
import pandas as pd
import matplotlib as mpl
import seaborn as sns

tips = sns.load_dataset("tips",)
print(tips.iloc[0:5])

sns.set()
sns.lmplot(x='total_bill', y='tip',
         hue='smoker', data=tips)
mpl.pyplot.show()
```

lmplot(x, y, data, hue=None, col=None, row=None, palette=None, col_wrap=None, height=5, aspect=1, markers='o', sharex=True, sharey=True, hue_order=None, col_order=None, row_order=None, legend=True, legend_out=True, x_estimator=None, x_bins=None, x_ci='ci', scatter=True, fit_reg=True, ci=95, n_boot=1000, units=None, order=1, logistic=False, lowess=False, robust=False, logx=False, x_partial=None, y_partial=None, truncate=False, x_jitter=None, y_jitter=None, scatter_kws=None, line_kws=None, size=None)

```
Plot data and regression model fits across a Fac
```

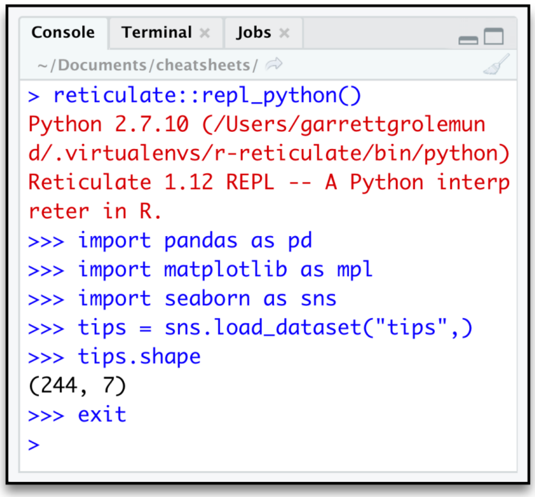
```
> reticulate::repl_python()
Python 2.7.10 (/Users/garrettgrolemund/.virtualenvs/r-reticulate/bin/python)
Reticulate 1.12 REPL -- A Python interpreter in R.
>>> import pandas as pd
>>>
```

A Python REPL opens in the console when you run Python code with a keyboard shortcut. Type **exit** to close.

Python REPL

A REPL (Read, Eval, Print Loop) is a command line where you can run Python code and view the results.

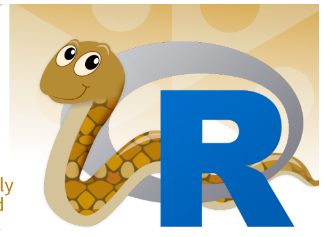
1. Open in the console with **repl_python**(), or by running code in a Python script with **Cmd + Enter** (**Ctrl + Enter**).
   - **repl_python**(module = NULL, quiet = getOption("reticulate.repl.quiet", default = FALSE)) Launch a Python REPL. Run **exit** to close. *repl_python()*
2. Type commands at **>>>** prompt
3. Press **Enter** to run code
4. Type **exit** to close and return to R console



```
> reticulate::repl_python()
Python 2.7.10 (/Users/garrettgrolemund/.virtualenvs/r-reticulate/bin/python)
Reticulate 1.12 REPL -- A Python interpreter in R.
>>> import pandas as pd
>>> import matplotlib as mpl
>>> import seaborn as sns
>>> tips = sns.load_dataset("tips",)
>>> tips.shape
(244, 7)
>>> exit
>
```

Configure Python

Reticulate binds to a local instance of Python when you first call **import**() directly or implicitly from an R session. To control the process, find or build your desired Python instance. Then suggest your instance to reticulate. **Restart R to unbind.**



Find Python

- **py_discover_config**() Return all detected versions of Python. Use **py_config** to check which version has been loaded. *py_config()*
- **py_available**(initialize = FALSE) Check if Python is available on your system. Also **py_module_available**, **py_numpy_module.** *py_available()*
- **virtualenv_list**() List all available virtualenvs. Also **virtualenv_root**(). *virtualenv_list()*
- **conda_list**(conda = "auto") List all available conda envs. Also **conda_binary**() and **conda_version**(). *conda_list()*

Create a Python env

- **virtualenv_create**(envname) Create a new virtualenv. *virtualenv_create("r-pandas")*
- **conda_create**(envname, packages = NULL, conda = "auto") Create a new Conda env. *conda_create("r-pandas", packages = "pandas")*

Install Packages

Install Python packages with R (below) or the shell:

pip install SciPy
conda install SciPy

- **py_install**(packages, envname = "r-reticulate", method = c("auto", "virtualenv", "conda"), conda = "auto", ...) Installs Python packages into a Python env named "r-reticulate". *py_install("pandas")*
- **virtualenv_install**(envname, packages, ignore_installed = FALSE) Install a package within a virtualenv. *virtualenv_install("r-pandas", packages = "pandas")*
- **virtualenv_remove**(envname, packages = NULL, confirm = interactive()) Remove individual packages or an entire virtualenv. *virtualenv_remove("r-pandas", packages = "pandas")*
- **conda_install**(envname, packages, forge = TRUE, pip = FALSE, pip_ignore_installed = TRUE, conda = "auto") Install a package within a Conda env. *conda_install("r-pandas", packages = "plotly")*
- **conda_remove**(envname, packages = NULL, conda = "auto") Remove individual packages or an entire Conda env. *conda_remove("r-pandas", packages = "plotly")*

Suggest an env to use

To choose an instance of Python to bind to, reticulate scans the instances on your computer in the following order, **stopping at the first instance that contains the module called by import().**

1. The instance referenced by the environment variable **RETICULATE_PYTHON** (if specified). **Tip: set in .Renviron file.**
   - **Sys.setenv**(RETICULATE_PYTHON = PATH) Set default Python binary. Persists across sessions! Undo with **Sys.unsetenv**. *Sys.setenv(RETICULATE_PYTHON = "/usr/local/bin/python")*
2. The instances referenced by **use_** functions if called before import(). Will fail silently if called after import unless **required = TRUE**.
   - **use_python**(python, required = FALSE) Suggest a Python binary to use by path. *use_python("/usr/local/bin/python")*
   - **use_virtualenv**(virtualenv = NULL, required = FALSE) Suggest a Python virtualenv. *use_virtualenv("~/myenv")*
   - **use_condaenv**(condaenv = NULL, conda = "auto", required = FALSE) Suggest a Conda env to use. *use_condaenv(condaenv = "r-nlp", conda = "/opt/anaconda3/bin/conda")*
3. Within virtualenvs and conda envs that carry the same name as the imported module. e.g. *~/anaconda/envs/nltk* for *import("nltk")*
4. At the location of the Python binary discovered on the system PATH (i.e. **Sys.which("python")**)
5. At customary locations for Python, e.g. **/usr/local/bin/python, /opt/local/bin/python...**

RStudio