Francesca System2 Code Samples

Frank Kovacs

9/20/2017

First, we must define the **path** to the `.py` file. This is best defined as a local path with the location of the file calling the command as the starting point. For example, if this file is located in the same folder as a "Scripts" folder that contains the program I wish to run (in this case we will call it `make_cov.py`), the path would be defined as follows:

```
path_to_py <- "Scripts/make_cov.py"
```

This variable will later be passed to the `system2` command as a string.

The goal of the `system2` command is to mimic the command line. For example, say I have the following script written in `Scripts/print_names.py`:

```
import string

names = ["Frank", "Alex", "Wenting", "Rebecca", "Genovese", "Francesca"]

for person in names:
  print("A person named %s in the array!" % person)

print("We have finished the script!")
```

If we were to run the script from the command line interface, we would run the following command:

```
python Scripts/print_names.py
```

```
## A person named Frank in the array!
## A person named Alex in the array!
## A person named Wenting in the array!
## A person named Rebecca in the array!
## A person named Genovese in the array!
## A person named Francesca in the array!
## We have finished the script!
```

To run such a command from native R, we may make use of the `system2` command. To illustrate this example, consider the output from the following example:

```
system2("python", args = c("Scripts/print_names.py"), stdout = TRUE)
```

```
## [1] "A person named Frank in the array!"
## [2] "A person named Alex in the array!"
## [3] "A person named Wenting in the array!"
## [4] "A person named Rebecca in the array!"
## [5] "A person named Genovese in the array!"
## [6] "A person named Francesca in the array!"
## [7] "We have finished the script!"
```