

Ruby | for MacAdmins



Nic Scott

Senior Apple Admin | Kenyon College

Professional Interest: ruby, scripting, automation

Personal Enjoyments: HBO, dry humor, IPA's

Slack: [@nic.scott](#)

Twitter: [@niclscott](#)

Github: <https://github.com/nlscott>

Email: scottnl@kenyon.edu

Blog: <https://redlinetech.wordpress.com>

Why Ruby?

Overview

1. Ruby Setup
2. Basic Ruby Concepts
3. Working Plist
4. Things Admins Can Do

Setup

Is Ruby installed?

What version do I have?

How do I run a script?

\$ ruby -v

ruby 2.0.0p648

\$which ruby

/usr/bin/ruby

\$irb

irb(main):001:0>

Ruby Setup

irb(main):001:0> puts "Hello"

Hello

=> nil

```
#!/usr/bin/ruby
```

```
puts "Hello"
```

Save file to Desktop as hello.rb

Open Terminal, type:

ruby /path_to_hello.rb

or

chmod +x /path_to_hello.rb

```
[nscott@robot:~$ /Users/nscott/Desktop/hello.rb
hello
```

Ruby Setup

Strings & Integers

a string in ruby is simply characters enclosed by quotes

a integer in ruby is whole number

```
> puts "Hello Mac Admins"
```

Hello Mac Admins

```
#convert all text to lowercase  
puts "Hello Mac Admins".downcase
```

hello mac admins

```
#convert all text to uppercase  
puts "Hello Mac Admins".upcase
```

HELLO MAC ADMINS

```
#concatenate strings with +  
puts "Hello " + "Mac " + "Admins "
```

Hello Mac Admins

```
#anything inside quotes is a string  
puts "3"
```

3

`#.class` will tell you what kind of object you have
`puts "3".class`

String

```
#list of methods for class  
puts String.instance_methods
```

| | | | | | | |
|----------|-----------------|-------------|-------------|--------------|-------------|--------------|
| eq? | setbyte | << | rstrip! | ascii_only? | methods | send |
| hash | byteslice | prepend | tr | unpack | singleton_m | public_send |
| casecmp | to_i | crypt | tr_s | encode | ethods | * |
| + | to_f | intern | delete | encode! | protected_ | respond_to |
| * | to_s | to_sym | squeeze | to_r | methods | ? |
| % | to_str | ord | count | to_c | private_met | extend |
| [] | inspect | include? | tr! | > | hods | display |
| []= | dump | start_with? | tr_s! | >= | public_met | method |
| insert | upcase | end_with? | delete! | < | hods | public_met |
| length | downcase | scan | squeeze! | <= | instance_va | hod |
| size | capitalize | ljust | each_line | between? | riables | define_singl |
| bytesize | swapcase | rjust | each_byte | nil? | isten_metho | eton_metho |
| empty? | upcase! | center | each_char | !~ | d | d |
| =~ | downcase! | sub | each_code | class | object_id | object_id |
| match | capitalize! | gsub | point | singleton_cl | to_enum | to_enum |
| succ | swapcase! | chop | sum | ass | enum_for | enum_for |
| succ! | hex | chomp | slice | clone | equal? | equal? |
| next | oct | strip | slice! | dup | ! | ! |
| next! | split | lstrip | partition | taint | != | != |
| upto | lines | rstrip | rpartition | tainted? | instance_ev | instance_ev |
| index | bytes | sub! | encoding | untaint | al | al |
| rindex | chars | gsub! | force_enco | untrust | instance_ex | instance_ex |
| replace | codepoints | chop! | ding | untrusted? | ec | ec |
| clear | reverse | chomp! | b | trust | __send__ | __send__ |
| chr | reverse! | strip! | valid_encod | freeze | __id__ | __id__ |
| getbyte | concat | lstrip! | ing? | frozen? | => nil | => nil |

Ruby Strings & Integers

#integers do not need quotes
puts 3

3

puts 3 + 5

8

`#.class` will tell you what kind of object you have
`puts 3.class`

Fixnum

```
#list of methods for class  
puts Fixnum.instance_methods
```

Core API

#to see examples

<https://ruby-doc.org>

- under "Core API", click on
your version of ruby

These are the API documents for the base classes and modules in the current stable release of Ruby 2.4.

- [2.4.1 core](#) - Core API docs for Ruby 2.4.1 This is the current official release.
- [2.4.0 core](#) - Core API docs for Ruby 2.4.0
- [2.3.4 core](#) - Core API docs for Ruby 2.3.4
- [2.3.3 core](#) - Core API docs for Ruby 2.3.3
- [2.3.2 core](#) - Core API docs for Ruby 2.3.2
- [2.3.1 core](#) - Core API docs for Ruby 2.3.1
- [2.3.0 core](#) - Core API docs for Ruby 2.3.0
- [2.2.7 core](#) - Core API docs for Ruby 2.2.7
- [2.2.6 core](#) - Core API docs for Ruby 2.2.6
- [2.2.5 core](#) - Core API docs for Ruby 2.2.5
- [2.2.4 core](#) - Core API docs for Ruby 2.2.4
- [2.2.3 core](#) - Core API docs for Ruby 2.2.3
- [2.2.2 core](#) - Core API docs for Ruby 2.2.2
- [2.2.1 core](#) - Core API docs for Ruby 2.2.1
- [2.2.0 core](#) - Core API docs for Ruby 2.2.0

- Scroll down to "Classes"
- manually find
- search box

| Classes | fix |
|-----------------------|-----|
| C ArgumentError | |
| C Array | |
| C BasicObject | |
| C Bignum | |
| C Binding | |
| C Class | |
| C ClosedQueueError | |
| M Comparable | |
| C Complex | |
| C Complex::compatible | |
| C ConditionVariable | |
| C Continuation | |
| C Data | |

[Home](#) [Classes](#) [Methods](#)

In Files

 numeric.c

Parent

Integer

Methods

#%
#&
#*
#**
#+
#-
#-@
#/

#<
#<<
#<=
#<=>
#==
#====
#>

Fixnum

Holds Integer values that can be represented in a native machine word (minus 1 bit). If any operation on a Fixnum exceeds this range, the value is automatically converted to a Bignum.

Fixnum objects have immediate value. This means that when they are assigned or passed as parameters, the actual object is passed, rather than a reference to that object.

Assignment does not alias Fixnum objects. There is effectively only one Fixnum object instance for any given integer value, so, for example, you cannot add a singleton method to a Fixnum. Any attempt to add a singleton method to a Fixnum object will raise a `TypeError`.

Public Instance Methods

fix % other → real

Returns `fix` modulo `other`.

See `Numeric#divmod` for more information.

Variables

Variables are used to store information



Local Variables
Constants
Global Variables
Class Variables
Instance Variables

Local Variables:
should start with a lower case letter or an underscore

```
name = "Joe"  
_name = "Bob"
```

Constants:

Should be used with all caps and meant for items
that will not change

APPFOLDER = "/Applications"

Variables

Global Variables:

begins with '\$' and has a global scope; meaning it can be accessed from anywhere within the program during runtime

```
$serial = "C02M50AGF6T6"
```

Instance Variables:

An instance variable has a name beginning with @, and its scope is confined to whatever object self refers to

```
@printer = "Ricoh 3005"
```

Class Variables:

A class variable is shared by all instances of a class and begins with '@@'

```
@@number_of_users = 20
```

Flow Control

Using conditional statements and loops

If Statement

```
1 |x = 3  
2  
3 if x == 3  
4   puts "x is 3"  
5 end  
6
```

```
1 x is 3  
2 [Finished in 0.3s]
```

If/Else Statement

```
1 x = 3
2
3 if x == 2
4   puts "x is 2"
5 else
6   puts "x is NOT 2"
7 end
8
```

```
1 x is NOT 2
2 [Finished in 0.3s]
```

Case Statement

```
1  
2 a = 5  
3  
4 case a  
5 when 5  
6   puts "a is 5"  
7 when 6  
8   puts "a is 6"  
9 else  
10  puts "a is neither 5, nor 6"  
11 end
```

```
1  
2 a is 5  
3 [Finished in 0.5s]
```

Until Loop

```
2 x = 5
3 until x == 0
4   x = x- 1
5   puts "This loop runs #{x} more times"
6 end
```

```
2 This loop runs 4 more times
3 This loop runs 3 more times
4 This loop runs 2 more times
5 This loop runs 1 more times
6 This loop runs 0 more times
7 [Finished in 0.3s]
```

While Loop

```
1  
2 x = 5  
3 while x > 0  
4   x = x - 1  
5   puts "This loop runs #{x} more times"  
6 end
```

```
1  
2 This loop runs 4 more times  
3 This loop runs 3 more times  
4 This loop runs 2 more times  
5 This loop runs 1 more times  
6 This loop runs 0 more times  
7 [Finished in 0.1s]
```

For Loop

```
1  
2 for x in 1..10  
3     puts x  
4 end
```

```
1 |  
2 1  
3 2  
4 3  
5 4  
6 5  
7 6  
8 7  
9 8  
10 9  
11 10
```

Each

```
2 numbers = [1,2,3,4,5]
3
4 numbers.each do |x|
5   puts x
6 end
```

```
2 1
3 2
4 3
5 4
6 5
7 [Finished in 0.3s]
```

Arrays

Arrays are ordered, integer-indexed collections of any object

Print objects in array

```
1  
2 fruit = ["apple", "orange", "banana"]  
3  
4 puts fruit  
5  
6
```

```
1  
2 apple  
3 orange  
4 banana  
5 [Finished in 0.1s]
```

Print objects in array by index

```
1  
2 fruit = ["apple", "orange", "banana"]  
3  
4 puts fruit[0]  
5
```

```
1  
2 apple  
3 [Finished in 0.3s]
```

Print objects index location

```
1  
2 fruit = ["apple", "orange", "banana"]  
3  
4 puts fruit.index("banana")  
5  
6
```

```
1  
2 2  
3 [Finished in 0.3s]
```

Add an object to an array

```
1  
2 fruit = ["apple", "orange", "banana"]  
3  
4 fruit.push("kiwi")  
5  
6 puts fruit
```

```
1  
2 apple  
3 orange  
4 banana  
5 kiwi  
6 [Finished in 0.3s]
```

Delete an object from an array

```
1  
2 fruit = ["apple", "orange", "banana", "kiwi"]  
3  
4 fruit.delete("kiwi")  
5  
6 puts fruit
```

```
1  
2 apple  
3 orange  
4 banana  
5 [Finished in 0.4s]
```

Print array in alphabetical order

```
1  
2 fruit = ["apple", "orange", "banana"]  
3  
4 puts fruit.sort  
5
```

```
1  
2 apple  
3 banana  
4 orange  
5 [Finished in 0.3s]
```

Print number of objects in an array

```
1  
2 fruit = ["apple", "orange", "banana"]  
3  
4 puts fruit.length  
5
```

```
1  
2 3  
3 [Finished in 0.3s]
```

Print only unique items in an array

```
1  
2 fruit = [  
3   "apple", "orange",  
4   "banana", "apple"]  
5  
6 puts fruit.uniq  
7
```

```
1  
2 apple  
3 orange  
4 banana  
5 [Finished in 0.1s]
```

Hashes

A Hash is a collection of key-value pairs

Hash example

```
1  
2 person = {  
3   "name" => "John",  
4   "age" => 40,  
5   "favorite food" => "Pizza"}  
6  
7  
8
```

```
1
```

Hash

Puppet example

```
1
2 user { 'test':
3   ensure      => 'present',
4   comment     => 'test',
5   gid         => '20',
6   groups      => ['admin'],
7   home        => '/Users/test',
8   iterations   => '40000',
9   shell        => '/bin/bash',
10  uid          => '502',
11 }
```

```
1
```

Hash

Print hash key and values

```
1
2 oses ={
3     "10.12" => "Sierra",
4     "10.11" => "El Capitan",
5     "10.10" => "Yosemite",
6     "10.9" => "Mavericks"}
7
8 oses.each do |key, value|
9     puts key + ": " + value
10 end
11
12
```

```
1
2 10.12: Sierra
3 10.11: El Capitan
4 10.10: Yosemite
5 10.9: Mavericks
6 [Finished in 0.3s]
```

Print hash key and values

```
1
2 oses ={
3     "10.12" => "Sierra",
4     "10.11" => "El Capitan",
5     "10.10" => "Yosemite",
6     "10.9" => "Mavericks"}
7
8 oses.each do | puppy_dogs, icecream_cones|
9     puts puppy_dogs + ":" + icecream_cones
10 end
```

```
1
2 10.12:Sierra
3 10.11:El Capitan
4 10.10:Yosemite
5 10.9:Mavericks
6 [Finished in 0.3s]
```

Hash

Check if a hash contains a specific key

```
1  
2 oses = {  
3   "10.12" => "Sierra",  
4   "10.11" => "El Capitan",  
5   "10.10" => "Yosemite",  
6   "10.9" => "Mavericks"}  
7  
8 puts oses.has_key? "Mountian Lion"
```

```
1  
2 false  
3 [Finished in 0.3s]
```

Check if a hash contains a specific key

```
1  
2 oses = {  
3   "10.12" => "Sierra",  
4   "10.11" => "El Capitan",  
5   "10.10" => "Yosemite",  
6   "10.9" => "Mavericks"}  
7  
8 puts oses.has_key? "10.9"
```

```
1  
2 true  
3 [Finished in 0.3s]
```

Print value for a key

```
1  
2 oses = {  
3     "10.12" => "Sierra",  
4     "10.11" => "El Capitan",  
5     "10.10" => "Yosemite",  
6     "10.9" => "Mavericks"}  
7  
8 puts oses["10.9"]  
9  
10
```

```
1  
2 Mavericks  
3 [Finished in 0.3s]
```

Add a key, value pair to a hash

```
1  
2 oses = {  
3     "10.12" => "Sierra",  
4     "10.11" => "El Capitan",  
5     "10.10" => "Yosemite",  
6     "10.9" => "Mavericks"}  
7  
8 oses["10.7"] = "Lion"  
9  
10 puts oses
```

```
1  
2 {"10.12"=>"Sierra",  
  "10.11"=>"El Capitan",  
  "10.10"=>"Yosemite",  
  "10.9"=>"Mavericks",  
  "10.7"=>"Lion"}  
3 [Finished in 0.1s]
```

Delete a key, value pair to a hash

```
1  
2 oses = {  
3     "10.12" => "Sierra",  
4     "10.11" => "El Capitan",  
5     "10.10" => "Yosemite",  
6     "10.9" => "Mavericks",  
7     "10.7" => "Lion"}  
8  
9 oses.delete("10.7")  
10  
11 puts oses
```

```
1  
2 {"10.12"=>"Sierra",  
  "10.11"=>"El Capitan",  
  "10.10"=>"Yosemite",  
  "10.9"=>"Mavericks"}  
3 [Finished in 0.1s]
```

Hash methods

```
1  
2 oses = {  
3   "10.12" => "Sierra",  
4   "10.11" => "El Capitan",  
5   "10.10" => "Yosemite",  
6   "10.9"  => "Mavericks"}  
7  
8 puts oses.empty?  
9 puts oses.length  
10 puts oses.keys  
11 puts oses.values  
12
```

```
1  
2 false  
3 4  
4 10.12  
5 10.11  
6 10.10  
7 10.9  
8 Sierra  
9 El Capitan  
10 Yosemite  
11 Mavericks  
12 [Finished in 0.3s]
```

Hash methods

```
1 |
2 oses ={
3     "10.12" => "Sierra",
4     "10.11" => "El Capitan",
5     "10.10" => "Yosemite",
6     "10.9"  => "Mavericks"}
7
8 puts oses.class
9 puts Hash.instance_methods
10
```

```
1
2 Hash
3 <
4 >
5 <=
6 >=
7 ==
8 []
9 []=
10 empty?
11 eql?
12 inspect
13 length
```

Methods

methods are used to bundle one or more repeatable statements

methods are very similar to functions in any other languages

should begin with a lowercase letter

should be defined before calling them

Method format

```
1
2 def method_name
3     code to execute....
4 end
5
6
```

Method example

```
1  
2 puts "Hello MacAdmins"  
3  
4 def psumac  
5 |   puts "Hello MacAdmins"  
6 end  
7  
8 psumac
```

```
1  
2 Hello MacAdmins  
3 Hello MacAdmins  
4 [Finished in 0.1s]
```

Method example

```
1  
2 def add  
3     puts 3 + 3  
4 end  
5  
6 add  
7 |
```

```
1  
2 6  
3 [Finished in 0.2s]
```

Method arguments

```
1  
2 def double(x)  
3   puts x * 2  
4 end  
5  
6 double(5)
```

```
1  
2 10  
3 [Finished in 0.3s]
```

Loops or iterators inside methods

```
1 def list_of_apps
2   Dir.entries("/Applications").each do |app|
3     if !app.start_with?(".")
4       puts app
5     end
6   end
7 end
8
9
10 list_of_apps
```

```
1 Adapter.app
2 Adobe InDesign CC 2015
3 Adobe Photoshop CC 2015.5
4 Atom.app
5 Audacity.app
6 BitBar.app
7 BlueGriffon.app
8 BlueStacks.app
9 Boom 2.app.zip
10 Brackets.app
11 Calculator.app
12
13
```

Working With Plist

CFPropertyList

a module to read, write, and manipulate both binary and XML property lists as defined by apple.

<https://rubygems.org/gems/CFPropertyList>

Checking for CFPropertyList

#open terminal and type
gem list

CFPropertyList (2.3.5)

Install CFPropertyList

#to install

```
sudo gem install CFPropertyList
```

#to update

```
sudo gem update CFPropertyList
```

Reading a plist with CFPropertyList

```
1
2 require 'cfpropertylist'
3
4 # loads plist into an hash named data
5 plistfile = "/Library/Preferences/com.apple.loginwindow.plist"
6 plist = CFPropertyList::List.new(:file => "#{plistfile}")
7 data = CFPropertyList.native_types(plist.value)
8
9 #print vaule
10 puts data['LoginwindowText']
11
12
13
```

```
1
2 For support contact Helpline x5700
3 [Finished in 0.2s]
```

#same as
defaults read "/Library/Preferences/com.apple.loginwindow.plist" LoginwindowText

Shelling out in Ruby

```
`defaults read "/Library/Preferences/com.apple.loginwindow.plist" LoginwindowText`
```

```
cmd = 'defaults read "/Library/Preferences/com.apple.loginwindow.plist" LoginwindowText'  
system(cmd)
```

Modify a plist with CFPropertyList

```
1 |
2 require 'cfpropertylist'
3
4 # loads plist into an hash named data
5 plistfile = "/Library/Preferences/com.apple.loginwindow.plist"
6 plist = CFPropertyList::List.new(:file => "#{plistfile}")
7 data = CFPropertyList.native_types(plist.value)
8
9 # modify hash value
10 data['LoginwindowText'] = "Call the Helpdesk for support"
11
12 # save data back to file
13 plist.value = CFPropertyList.guess(data)
14 plist.save("#{plistfile}", CFPropertyList::List::FORMAT_BINARY)
```

#same as

sudo defaults write /Library/Preferences/com.apple.loginwindow.plist LoginwindowText "Call the Helpdesk for support"

What Can MacAdmins Do With Ruby?

List users on machine

```
1  #!/usr/bin/ruby
2
3  def list_of_users
4      users=[]
5      for username in Dir.entries("/Users")
6          if !username.start_with?(".")
7              users.push(username)
8          end
9      end
10     return users
11 end
12
13 puts "Total Users: #{list_of_users.length}"
14 puts list_of_users
15
```

```
1
2  Total Users: 5
3  faculty
4  jsmith
5  nscott
6  Shared
7  student
8
```

List users in local admin group

```
1  #!/usr/bin/ruby
2
3
4  def local_admins
5      list_of_users_tocheck = []
6      for x in Dir.entries("/Users")
7          if !x.start_with?(".")
8              list_of_users_tocheck.push(x)
9          end
10     end
11
12    localadmins = []
13    for x in list_of_users_tocheck
14        if `dsmemberutil checkmembership -U #{x} -G admin 2>/dev/null`.chomp == "user is a member of the group"
15            localadmins.push(x)
16        end
17    end
18    puts "Local Admins:"
19    return localadmins
20 end
21
22
23 def listoflocaladmins
24     listoflocaladmins =`dscacheutil -q group -a name admin`.split(":")
25     listoflocaladmins=listoflocaladmins[4].strip.split(" ")
26     puts "Local Admins:"
27     return listoflocaladmins
28 end
29
30 puts listoflocaladmins
```

```
1 Local Admins:
2 root
3 nscott
4 faculty
5
6
```

List of apps user installed after deployment

```
1 #!/usr/bin/ruby
2
3 base_apps = [ "App Store.app", "Automator.app", "Calculator.app",
4 "Calendar.app", "Chess.app", "Contacts.app", "Dashboard.app",
5 "Dictionary.app", "DVD Player.app", "FaceTime.app", "Font Book.app",
6 "Game Center.app", "Google Chrome.app", "iBooks.app", "Image Capture.app",
7 "iTunes.app", "Launchpad.app", "Mail.app", "Maps.app", "Messages.app",
8 "Mission Control.app", "Notes.app", "Photo Booth.app", "Photos.app",
9 "Preview.app", "QuickTime Player.app", "Reminders.app", "Safari.app",
10 "Stickies.app", "System Preferences.app", "TextEdit.app", "Time Machine.app",
11 "Utilities", "VLC.app" ]
12
13 users_apps = []
14
15 =====
16 #add apps installed by user to user_apps
17
18 Dir.entries("/Applications").each do | appname |
19   if !appname.start_with?(".")
20     if !base_apps.include?(appname)
21       users_apps.push(appname)
22     end
23   end
24 end
25
26 puts "Number of Apps Installed: #{users_apps.length}"
27 puts users_apps
28
```

```
1 Number of Apps Installed: 52
2 Adapter.app
3 Atom.app
4 Audacity.app
5 BitBar.app
6 BlueGriffon.app
7 BlueStacks.app
8 Boom 2.app.zip
9 Camtasia 2.app
10 Dash.app
11 Dropbox.app
12 Firefox.app
13 GarageBand.app
14 GeekTool.app
15 GitHub Desktop.app
16 Google Drive.app
17 Keynote.app
18 LastPass.app
19 Microsoft Excel.app
20 Microsoft OneNote.app
21 Microsoft Word.app
22 MusicManager.app
23 Numbers.app
24 OverSight.app
25 Packages.app
26 Pages.app
27 Platypus.app
28 Skype.app
29 Slack.app
30 SourceTree.app
31 Sublime Text.app
32 Suspicious Package.app
33 |
```

List apps in a users dock

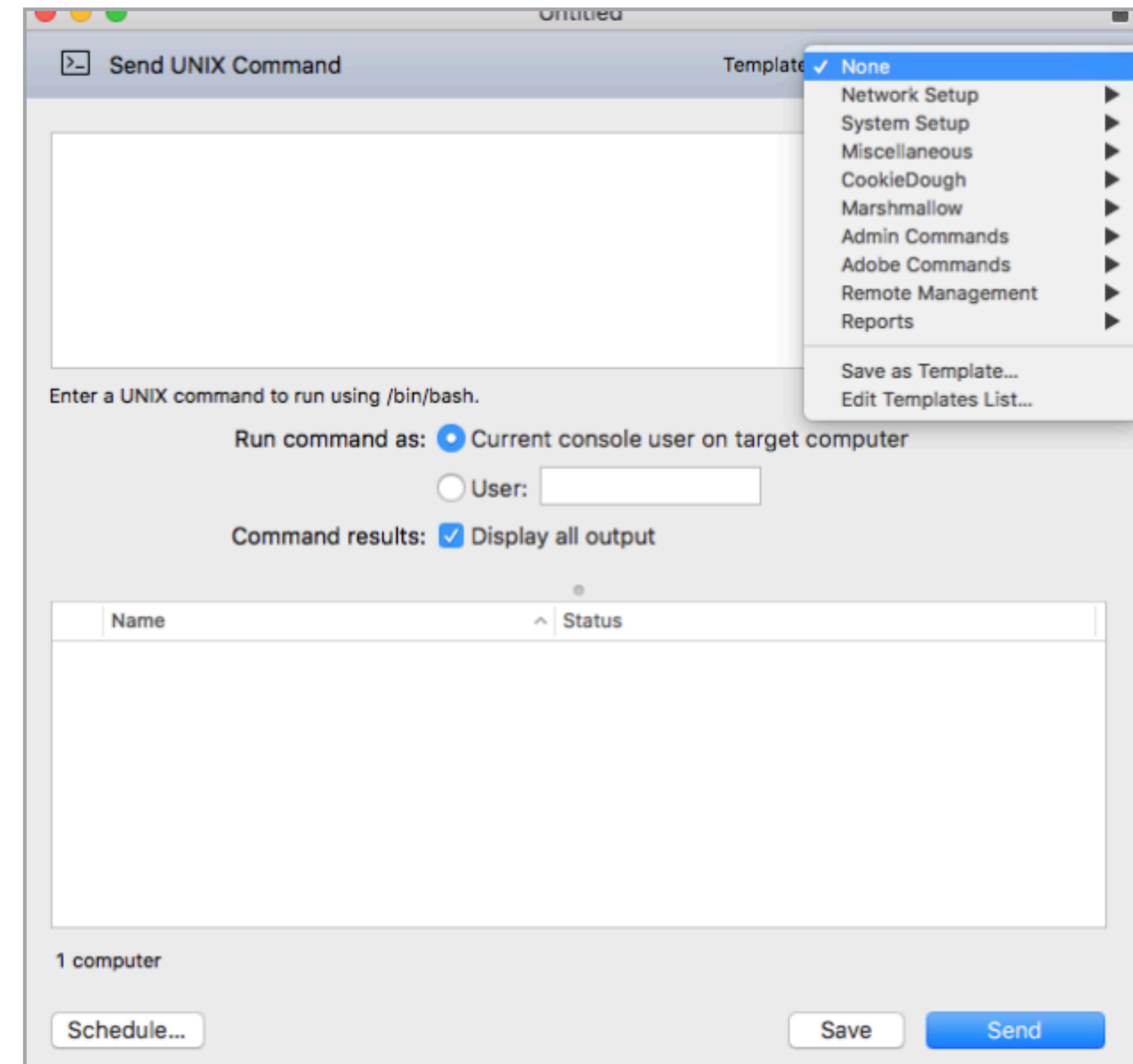
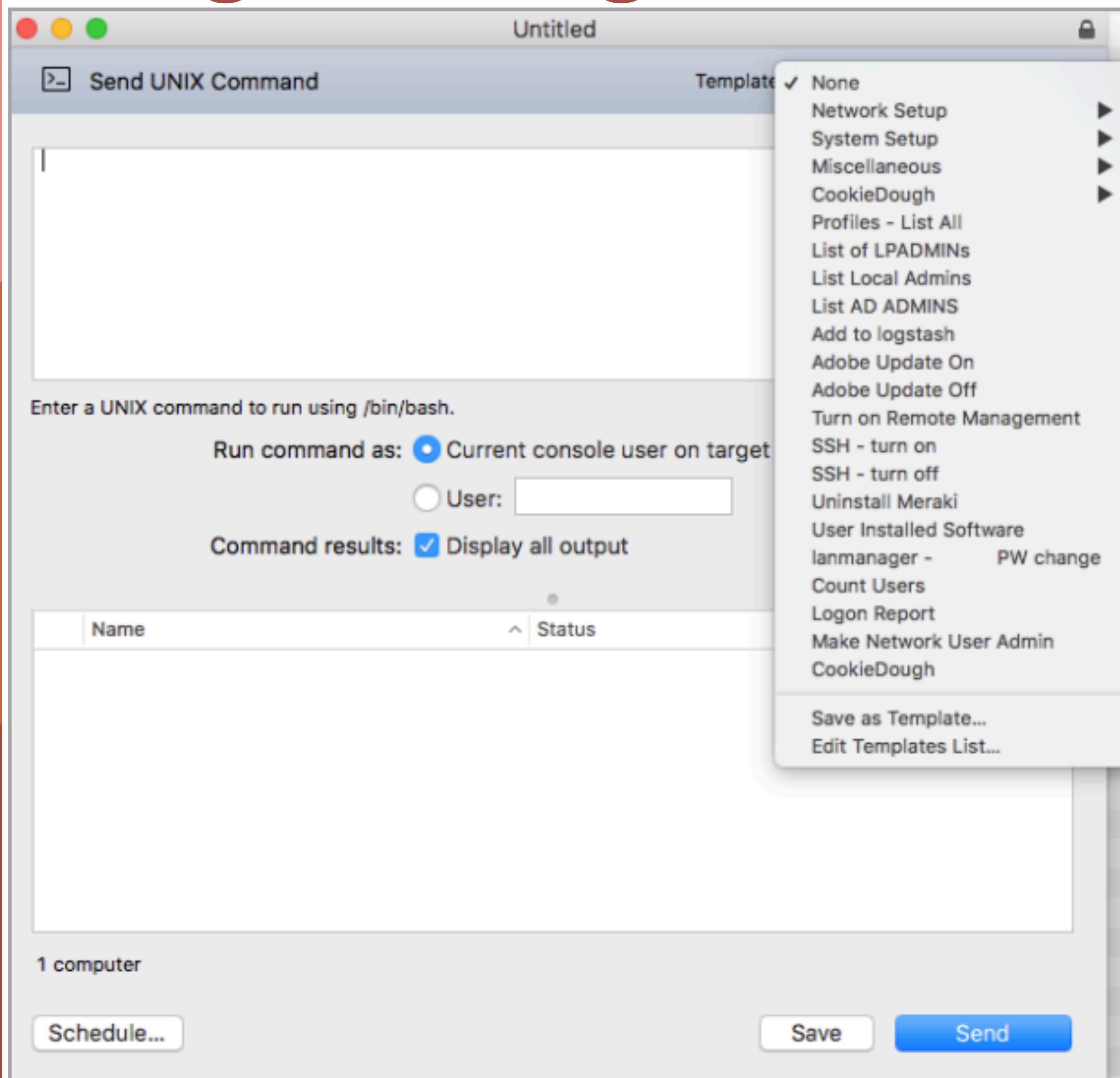
```
1 #!/usr/bin/ruby
2
3 require 'CFPropertyList'
4 require 'etc'
5
6 CURRENTUSER=Etc.getlogin
7
8 #=====
9 #list the aspps in a users dock
10 def dock_apps(user = CURRENTUSER)
11
12   plist = CFPropertyList::List.new(:file => "/Users/#{user}/Library/Preferences/com.apple.dock.plist")
13   results=CFPropertyList.native_types(plist.value)
14   apps=[]
15   for key, value in results['persistent-apps']
16     for key, value in key
17       if value.class == Hash
18         for x, y in value
19           if x == "file-label"
20             apps.push(y)
21           end
22         end
23       end
24     end
25   end
26   return apps
27 end
28
29 #=====
30 #default is current users dock
31 puts dock_apps
32
```

```
1 Messages
2 Firefox
3 Google Chrome
4 Safari
5 Notes
6 Reminders
7 Sublime Text
8 Atom
9 Dash
10 BitBar
11 Slack
12 SourceTree
13 GitHub Desktop
14
15 [Finished in 0.5s]
```



```
1  #!/usr/bin/ruby
2
3  DOCKUTIL="/usr/local/bin/dockutil"
4  APPS="/Applications"
5
6  @add_app=["Launchpad.app", "App Store.app", "Safari.app", "Google Chrome.app", "Google Drive.app", "Mail.app", "Contacts.app",
7    "Calendar.app", "Notes.app", "Reminders.app", "Maps.app", "Photos.app", "Messages.app", "FaceTime.app",
8    "iTunes.app", "Microsoft Excel.app", "Microsoft PowerPoint.app", "Microsoft Word.app", "blender.app",
9    "SketchUp 2017/SketchUp.app", "Adobe Photoshop CC 2017/Adobe Photoshop CC 2017.app"]
10
11 @folder='~/Downloads'
12
13 def check_for_dockutil
14   if !File.exist?("/usr/local/bin/dockutil")
15     abort "Dockutil is not installed"
16   end
17 end
18
19 def set_user_dock
20   removeallapps = "#{DOCKUTIL} --remove all --allhomes --no-restart"
21   system(removeallapps)
22
23   @add_app.each do |appname|
24     addapp="#{DOCKUTIL} --add '#{APPS}/#{appname}' --allhomes --no-restart"
25     system(addapp)
26   end
27
28   if !@folder.empty?
29     restart="#{DOCKUTIL} --add '#{@folder}' --view grid --display folder --allhomes --no-restart"
30     system(restart)
31   end
32 end
33
34 def set_user_template
35   removeallapps = "#{DOCKUTIL} --remove all --no-restart '/System/Library/User Template/English.lproj'"
36   system(removeallapps)
37
38   @add_app.each do |appname|
39     addapp="#{DOCKUTIL} --add '#{APPS}/#{appname}' --no-restart '/System/Library/User Template/English.lproj'"
40     system(addapp)
41   end
42 end
43
44 #run commands
45 check_for_dockutil
46 set_user_dock
47 set_user_template
48
```

Organizing Unix Commands in ARD



Organizing Unix Commands in ARD

| Key | Type | Value |
|----------|------------|-----------|
| Root | Array | (7 items) |
| ► Item 0 | Dictionary | (2 items) |
| ► Item 1 | Dictionary | (2 items) |
| ► Item 2 | Dictionary | (2 items) |
| ► Item 3 | Dictionary | (2 items) |
| ► Item 4 | Dictionary | (2 items) |
| ► Item 5 | Dictionary | (2 items) |
| ► Item 6 | Dictionary | (2 items) |

| Key | Type | Value |
|------------|------------|-------------|
| Root | Array | (7 items) |
| ► Item 0 | Dictionary | (2 items) |
| name | String | CookieDough |
| state | Array | (4 items) |
| ► Item 0 | Dictionary | (2 items) |
| name | String | list_apps |
| state | Dictionary | (4 items) |
| outputMode | Boolean | YES |
| script | String | RUBY |
| user | String | root |
| userSelect | Number | 1 |
| ► Item 1 | Dictionary | (2 items) |
| ► Item 2 | Dictionary | (2 items) |
| ► Item 3 | Dictionary | (2 items) |
| ► Item 1 | Dictionary | (2 items) |
| ► Item 2 | Dictionary | (2 items) |
| ► Item 3 | Dictionary | (2 items) |
| ► Item 4 | Dictionary | (2 items) |
| ► Item 5 | Dictionary | (2 items) |
| ► Item 6 | Dictionary | (2 items) |

Organizing Unix Commands in ARD

The screenshot shows the ARD application window. On the left, a script editor window displays a Ruby script for managing Unix commands. On the right, a sidebar menu lists various command categories: Network Setup, System Setup, Miscellaneous, CookieDough, Marshmallow, Admin Commands, Adobe Commands, Remote Management, and Reports. A 'None' option is selected at the top. At the bottom of the sidebar, there are 'Save as Template...' and 'Edit Templates List...' options.

```
class Command
  attr_reader :foldername, :commandname, :outputmode, :script, :user, :userselect

  def initialize(foldername, commandname, outputmode, script, user, userselect)
    @foldername = foldername
    @commandname = commandname
    @outputmode = outputmode
    @script = script
    @user = user
    @userselect = userselect
  end

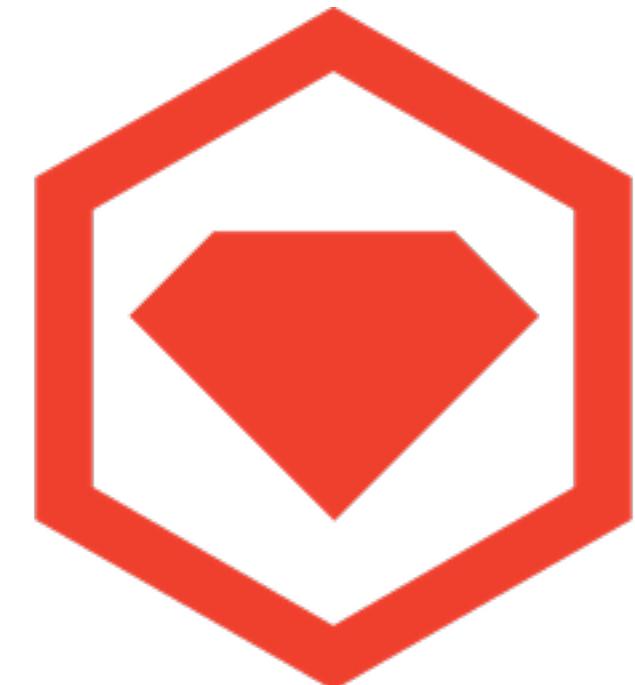
  #
  # Methods

  #print folder names
  def self.list_folders
    folders=[]
    for x in @results
      folders.push(x['name'])
    end
    return folders
  end

  #find index of folder
  def self.folder_index(folder)
    indexnumber=nil
    for x in @results
      if x['name'] == folder
        indexnumber = @results.index(x)
      end
    end
    if indexnumber.nil?
      return "Folder does not exist"
    else
      return indexnumber
    end
  end
end
```

ARD.create_empty_folder("folder1")

ARD.move_command("Test Command", "Folder 1")



Ruby Gems

marshmallow 0.1.3

A collection of macOS facts and useful information for MacAdmins

cookiedough 0.1.1

A Ruby gem for macOS machines to report on Applications

panes 0.1.2

A gem to manage System Preference Panes. Enable or disable access to individual panes.

ard 0.1.0

A gem to create and organize commands in Apple Remote Desktop

```
1  
2 require "Marshmallow"  
3  
4 puts Marshmallow.system_launchagents
```

```
1  
2 com.adobe.AdobeCreativeCloud.plist  
3 com.google.keystone.agent.plist  
4 com.oracle.java.Java-Updater.plist  
5 [Finished in 0.4s]
```

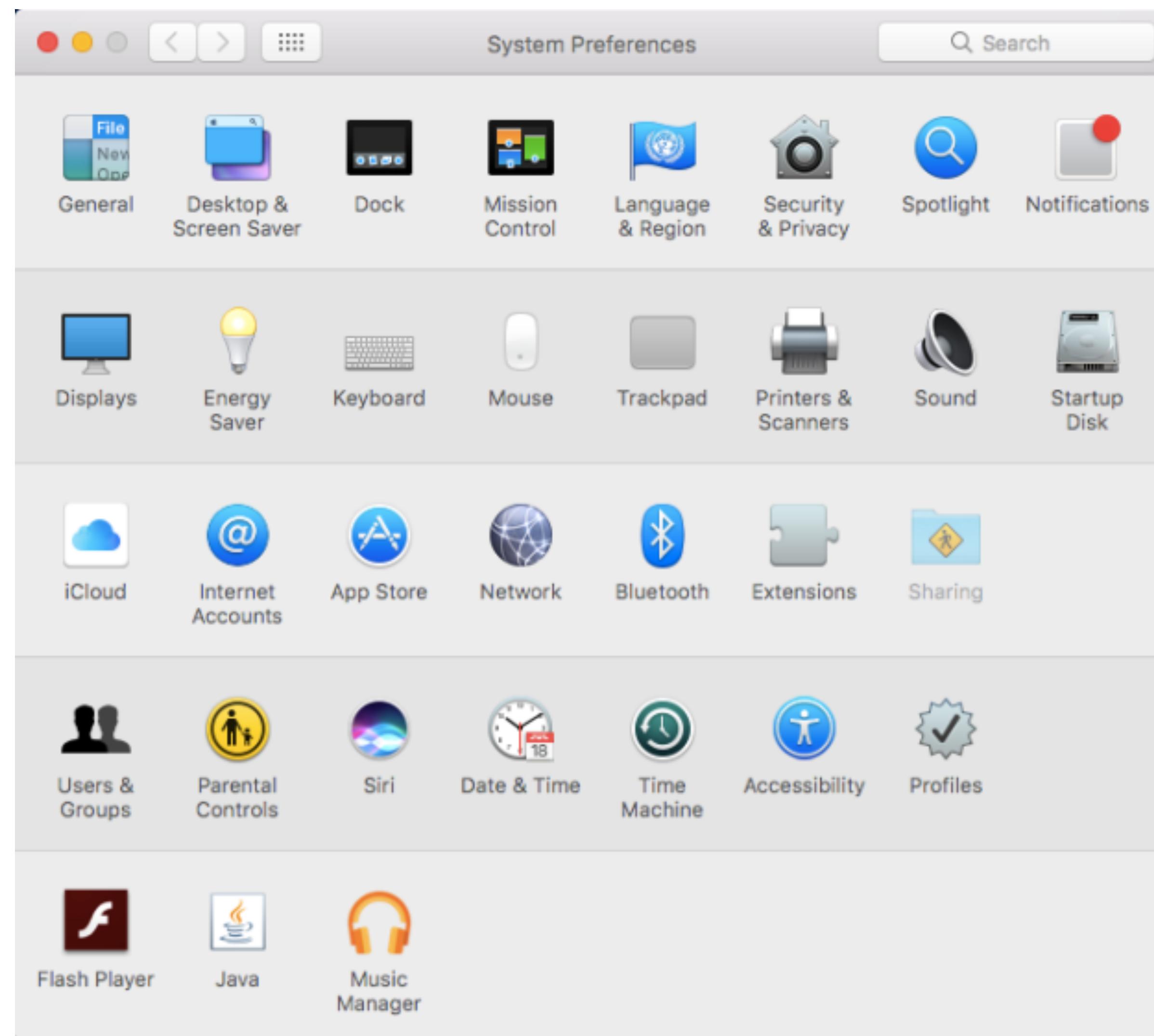
```
1  
2 require "CookieDough"  
3  
4 puts CookieDough.dock_apps("nscott")  
5  
6
```

```
1  
2 Messages  
3 Firefox  
4 Google Chrome  
5 Safari  
6 iTunes  
7 Notes  
8 Reminders  
9 Sublime Text  
10 Dash  
11 BitBar  
12 Slack  
13 Atom  
14 SourceTree  
15 GitHub Desktop  
16 Skype  
17 [Finished in 0.2s]
```

```
1  
2 require "CookieDough"  
3  
4 puts CookieDough.all_app_versions  
5  
6
```

```
1  
2 ...  
3 Microsoft OneNote.app = 15.33  
4 Mission Control.app = 1.2  
5 Monitor.app = 1.0.6  
6 MusicManager.app = 1.0  
7 NoMAD.app = 1.0.3  
8 Notes.app = 4.3.1  
9 Numbers.app = 4.1.1  
10 OverSight.app = 1.0.0  
11 Pages.app = 6.1.1  
12 Photo Booth.app = 8.0  
13 Photos.app = 2.0  
14 Platypus.app = 5.1  
15 Postgres.app = 2.0.1  
16 Preview.app = 9.0  
17 QuickTime Player.app = 10.4  
18 Reminders.app = 4.0  
19 ...
```

Disable System Preference Panes



Panes.Disable("com.apple.preferences.sharing")

What Else?

Finder

Printers

DeployStudio Reporting

collecting or filtering data

References & Resources

Alleviate the Apprehension of Coding in Ruby: <https://goo.gl/B4SQ4t>

Ruby Docs: <http://ruby-doc.org/>

#ruby in Macadmins Slack

CodeCademy: <https://www.codecademy.com/>

LaunchSchool: <https://launchschool.com/books/ruby/read>

Ruby Gems: <https://rubygems.org/>

References & Resources

REDline: <https://redlinetech.wordpress.com>

GEMS: https://rubygems.org/profiles/nic_scott

GitHub: <https://github.com/nlscott>



Ruby

Learn Ruby. Do Something Cool.