Fox Documentation

Release 0.1.0

Daniel Kertesz

Dec 14, 2021

Contents:

Fox is a Python package for creating quick and dirty server automation scripts. For example to jump into the late nineties:

```
from fox.conf import env
from fox.api import run, sudo

env.host_string = "server.example.com"
env.sudo_password = "very secret"

run("./configure --with-prefix=/90s", cd="/code/project")
sudo("make install", cd="/code/project")
```

CHAPTER 1

API

1.1 Configuration Objects

class fox.conf.**Environment**

Fox is configured through the global variable *env* which is an instance of *Environment*.

host_string = None
The hostname of the remote server that will be used in all of the *run()*, *sudo()* and all of the other remote commands.

port = None
The remote port.

private_key = None
The path to a OpenSSH private key.

ssh_config_path = '/home/docs/.ssh/config'
Set the path to the OpenSSH configuration file.

sudo_password = None
Set the password for the *sudo()* commands.

sudo_prompt = 'sudo password:'
The prompt for sudo commands (do not change!).

term_size = (80, 24)
The size of the emulated terminal when a *pty* is requested.

term_type = 'vt100'
The terminal type to emulate when a *pty* is requested.

use_ssh_config = True
Set to *True* to enable the loading of *~/.ssh/config*.

username = None
The remote username.

fox.conf.**env = <fox.conf.Environment object>**
Global configuration object.

1.2 API Methods

fox.api.**run**(*command*, *pty=False*, *cd=None*, *environ=None*, *echo=True*) → fox.utils.CommandResult
Run a command on the current *env.host_string* remote host.

Parameters

- **command** – the command line string to execute.
- **pty** – wether to request a remote pty.
- **cd** – the optional name of the directory where the command will be executed.
- **environ** – an optional dictionary containing environment variables to set when

executing the command. :param echo: set to *False* to hide the output of the command.

fox.api.**run_concurrent**(*hosts*, *command*, *limit=0*)
Execute *command* on *hosts* concurrently.

Parameters

- **hosts** – a list of hosts where to run *command*.
- **command** – the command line string to execute.
- **limit** – limit the concurrent execution to *limit* hosts; set to *0* to execute on all the

hosts at once.

fox.api.**sudo**(*command*, *pty=False*, *cd=None*, *environ=None*, *echo=True*) → fox.utils.CommandResult
Run a command on the current env.host_string remote host with sudo

Parameters

- **command** – the command line string to execute.
- **pty** – wether to request a remote pty.
- **cd** – the optional name of the directory where the command will be executed.
- **environ** – an optional dictionary containing environment variables to set when

executing the command. :param echo: set to *False* to hide the output of the command.

fox.api.**get**(*remotefile*, *localfile*)
Download a file from the remote server.

Parameters

- **remotefile** – the path to the remote file to download.
- **localfile** – the local path where to write the downloaded file.

fox.api.**put**(*localfile*, *remotefile*)
Upload a local file to a remote server.

Parameters

- **localfile** – the path of the local file to upload.
- **remotefile** – the path where to write the file on the remote server.

fox.api.**read**(*remotefile*) → bytes
Read the contents of a remote file.

Parameters **remotefile** – the path of the remote file to read.

This is useful when you just want to read the contents of a remote file without downloading it.

fox.api.**file_exists**(*remotefile*) → bool
Check if a file exists on the remote server.

Parameters **remotefile** – the path of the remote file that will be checked.

fox.api.**local**(*command*, *cd=None*, *environ=None*, *env_inherit=True*) → fox.utils.CommandResult
Execute *command* on the local machine.

Parameters

- **command** – the command line string to execute.
- **cd** – the optional name of the directory where the command will be executed.
- **environ** – an optional dictionary containing environment variables to set when

executing the command. :param env_inherit: set to *False* when you also specify *env* to execute the process in a new blank environment.

1.3 Connection Object

class fox.connection.**Connection**(*hostname: str*, *username: str*, *port: int*, *private_key=None*, *password: Optional[str] = None*, *agent_path: Optional[str] = None*, *tunnel: Optional[str] = None*, *nickname: Optional[str] = None*)
A SSH connection to a remote server.

Parameters

- **hostname** – hostname of the remote server.
- **username** – the username used to log into the remote server.
- **port** – the optional port for connecting to the remote server (default: 22).
- **private_key** – the optional path to a OpenSSH private key.
- **password** – the optional password used to authenticate to the remote server.
- **agent_path** – the optional path to a OpenSSH agent socket.
- **tunnel** – the optional hostname of another server that will be used as tunnel.
- **nickname** – the hostname of the server as passed on the command line (could be different from the real hostname configured in *~/.ssh/config*).

disconnect()
Close the SSH connection to the server.

file_exists(*remotefile*) → bool
Check if a file exists on the remote server.

Parameters **remotefile** – the path of the remote file that will be checked.

get(*remotefile*, *localfile*)
Download a file from the remote server.

Parameters

- **remotefile** – the path to the remote file to download.
- **localfile** – the local path where to write the downloaded file.

put(*localfile*, *remotefile*)
Upload a local file to a remote server.

Parameters

- **localfile** – the path of the local file to upload.
- **remotefile** – the path where to write the file on the remote server.

read(*remotefile*) → bytes
Read the contents of a remote file.

Parameters **remotefile** – the path of the remote file to read.

This is useful when you just want to read the contents of a remote file without downloading it.

run(*command*, *pty=True*, *cd=None*, *environ=None*, *echo=True*) → fox.utils.CommandResult
Execute a command on the remote server.

Parameters

- **command** – the command line string to execute.
- **pty** – wether to request a remote pty.
- **cd** – the optional name of the directory where the command will be executed.
- **environ** – an optional dictionary containing environment variables to set when executing the command.
- **echo** – set to *False* to hide the output of the command.

sudo(*command*, *pty=True*, *cd=None*, *environ=None*, *echo=True*) → fox.utils.CommandResult
Execute a command with sudo on the remote server.

Parameters

- **command** – the command line string to execute.
- **pty** – wether to request a remote pty.
- **cd** – the optional name of the directory where the command will be executed.
- **environ** – an optional dictionary containing environment variables to set when executing the command.
- **echo** – set to *False* to hide the output of the command.

1.4 Cluster Object

class fox.cluster.**Cluster**(**hosts*)
Cluster mode.

Run a command on several hosts in parallel.

TOOD: - canary run (do a canary run on the first host before doing the remanining) - exit after % of hosts failed

fox.cluster.**connect_pipes**(*source*, *source_command*, *destination*, *destination_command*)
Connects processes on two connections with a pipe

Pipe stdout and stderr from a command executed on a source connection to stdin of a process on a destination connection.

1.5 SSHConfig Object

class fox.sshconfig.**SSHConfig**
Parse a OpenSSH configuration file and lookup SSH options for connecting to a given host.

load(*filename: str*)
Load and parse a OpenSSH configuration file.

lookup(*nickname: str*) → Dict[str, Any]
Lookup SSH options for connecting to the server *nickname*.

exception fox.sshconfig.**Error**
An error encountered while parsing a OpenSSH configuration file

1.6 CommandResult Object

CommandResult objects are returned by all the *fox.api.run()*, *fox.api.sudo()*, *fox.api.local()* functions and their corresponding methods in the *fox.connection.Connection* class and can be used to inspect the results of the execution of a command.

class fox.utils.**CommandResult**(*command: str*, *actual_command: str*, *exit_code: int*, *stdout: str*, *stderr: str*, *hostname: str*, *sudo: bool = False*)
Use the *CommandResult.stdout* and *CommandResult.stderr* attributes to inspect *stdout* and *stderr* of the process.

actual_command = None
The actual command that was executed, including all the *cd* and *env* prefixes.

command = None
The command that was executed.

exit_code = None
The exit code of the executed command.

hostname = None
The hostname of the server where the command was executed.

stderr = None
The (partial) *stderr* output of the executed command.

stdout = None
The (partial) *stdout* output of the executed command.

sudo = False
Wether the command was executed with sudo.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

f

Index