Carnival

нояб. 24, 2021

Содержимое:

Глава 1

Оборудование (Host)

Объект, представляющий единицу оборудования.

Carnival не предоставляет никаких сложных абстракций для работы с группами хостов, подразумевая что вы будете использовать встроенные коллекции python и организуете работу так, как будет удобно для вашей задачи. В простом случае, можно передавать хосты прямо в коде файла *carnival_tasks.py*. В более сложных, создать списки в отдельном файле, например *inventory.py*

`class carnival.hosts.base.Host(`***context*`)`
Базовый класс для хостов

`addr = None`
Адрес хоста

`connect()` → carnival.hosts.base.Connection
Создать конект с хостом

`class carnival.hosts.base.Result(`*return_code: int*, *ok: bool*, *stdout: str*, *stderr: str*`)`
Результат выполнения команды

1.1 Базовые типы

`class carnival.hosts.base.Host`
Базовый класс для хостов

`addr = None`
Адрес хоста

`connect()` → carnival.hosts.base.Connection
Создать конект с хостом

`class carnival.hosts.base.Connection`

`host = None`
Хост с которым связан конект

`run`(*command: str*, *hide: bool = False*, *warn: bool = True*, *cwd: Optional[str] = None*) → carnival.hosts.base.Result
Запустить команду

Параметры

- `command` – Команда для запуска
- `hide` – Скрыть вывод команды
- `warn` – Вывести stderr
- `cwd` – Перейти в папку при выполнении команды

1.2 Локалхост

`class carnival.hosts.local.LocalHost`
Локальный хост, работает по локальному терминалу

`__init__`(***context*) → None

Параметры `context` – Контекст хоста

`addr = 'local'`
Адрес хоста, всегда *local*

`connect()` → carnival.hosts.local.LocalConnection
Создать конект с хостом

`class carnival.hosts.local.LocalConnection`

`run`(*command: str*, *hide: bool = False*, *warn: bool = True*, *cwd: Optional[str] = None*) → carnival.hosts.base.Result
Запустить команду

Параметры

- `command` – Команда для запуска
- `hide` – Скрыть вывод команды
- `warn` – Вывести stderr
- `cwd` – Перейти в папку при выполнении команды

1.3 SSH

`class carnival.hosts.ssh.SshHost`
SSH хост

`__init__`(*addr: str*, *ssh_user: Optional[str] = None*, *ssh_password: Optional[str] = None*, *ssh_port: int = 22*, *ssh_gateway: Optional[SshHost] = None*, *ssh_connect_timeout: int = 10*, *missing_host_key_policy: Type[<sphinx.ext.autodoc.importer._MockObject object at 0x7fd9798a3220>] = <sphinx.ext.autodoc.importer._MockObject object>*, ***context*)

Параметры

- `addr` – Адрес сервера
- `ssh_user` – Пользователь SSH

- `ssh_password` – Пароль SSH
- `ssh_port` – SSH порт
- `ssh_connect_timeout` – SSH таймаут соединения
- `ssh_gateway` – Gateway
- `context` – Контекст хоста

`addr = None`
Домен либо ip хоста

`connect()` → carnival.hosts.ssh.SshConnection
Создать конект с хостом

`class carnival.hosts.ssh.SshConnection`

`run(`*command: str*, *hide: bool = False*, *warn: bool = True*, *cwd: Optional[str] = None*`)` → carnival.hosts.base.Result
Запустить команду

Параметры

- `command` – Команда для запуска
- `hide` – Скрыть вывод команды
- `warn` – Вывести stderr
- `cwd` – Перейти в папку при выполнении команды

Глава 2

Шаги (Step)

class carnival.Step

Объект, предназначенный для выполнения группы команд с какой-то целью. Вызывается из класса *carnival.Task* для выполнения команд (*carnival.cmd*) на определенных хостах.

Может требовать наличие определенных контекстных переменных для работы, указав их в аргументах конструктора, а в задаче (Task) передать нужные аргументы в конструктор.

Может вернуть значение для дальнейшего использования.

```
>>> ...
>>> class DiskUsage(Step):
>>>     def __init__(self, disk_name: str):
>>>         self.disk_name = disk_name
>>>
>>>     def run(self, c: Connection):
>>>         ...
```

run(*c: Connection*) → Any

Метод который нужно определить для выполнения команд

Параметры c – Соединение с хостом для выполнения шага

validate(*c: Connection*) → None

Валидатор шага, запускается перед выполнением Должен выкидывать .StepValidationError в случае ошибки

Параметры host – На котором будет выполнен шаг

Исключение StepValidationError – в случае ошибок валидации

```
>>> from carnival.exceptions import StepValidationError
>>> ...
>>> def validate(self, c: "Connection") -> None:
>>>     raise StepValidationError("Step validation is not implemented")
```

Глава 3

Задача (Task)

class carnival.TaskBase(*no_validate: bool*)

Задача это единица выполнения одного или нескольких шагов на определенных хостах.

Именование задач.

Полное имя задачи состоит из двух частей. <module_name>.<name>. carnival автоматически генерирует имена задач из этих частей, но есть возможность управлять этим вручную, используя два атрибута класса Task.

name: module_name:

```
>>> class CheckDiskSpace(TaskBase):
>>>     help = "Print server root disk usage"
>>>
>>>     def run(self) -> None:
>>>         with my_server.connect() as c:
>>>             cmd.cli.run(f"df -h /", hide=False)
```

call_task(*task_class: Type[TaskBase]*) → Any

Запустить другую задачу Возвращает результат работы задачи

help = ''

Строка помощи при вызове carnival help

module_name = None

имя модуля. если назначить пустую строку, полное имя будет включать только название задачи.

name = ''

название задачи. если не определено имя будет сгенерировано автоматически.

run() → None

Реализация выполнения задачи

validate() → List[str]

Хук для проверки валидности задачи перед запуском, не вызывается автоматически

3.1 Задачи с шагами

class carnival.Task(*no_ validate: bool*)

Запустить шаги *steps* на хостах *hosts*

```
>>> class InstallPackages(StepsTask):
>>>     help = "Install packages"
>>>
>>>     hosts = [my_server]
>>>     steps = [InstallStep(my_server.context['packages'])]
```

get_steps(*host: carnival.hosts.base.Host*) → List[carnival.step.Step]

Список шагов в порядке выполнения

hosts = None

Список хостов для выполнения шагов

run() → None

Реализация выполнения задачи

validate() → List[str]

Хук для проверки валидности задачи перед запуском, проверяет примеримость контекста хостов на шагах

3.2 Встроенные задачи

carnival имеет встроенные задачи для удобства использования

class carnival.internal_tasks.Help(*no_ validate: bool*)

Показать список доступных задач

class carnival.internal_tasks.Validate(*no_ validate: bool*)

Запустить валидацию доступых задач и напечатать список ошибок

Глава 4

Интерфейс командной строки (cli)

carnival.cli.except_hook(*type: Type[Any], value: Any, traceback: Any*) → None

carnival.cli.is_completion_script(*complete_var: str*) → bool

carnival.cli.main() → int

```
>>> $ poetry run python -m carnival --help
>>> Usage: python -m carnival [OPTIONS] {help|test}...
>>> Options:
>>> --debug        Turn on debug mode
>>> --no_validate  Disable step validation
>>> --help         Show this message and exit.
```

Глава 5

Шаблоны jinja2 (templates)

`carnival.templates.render`(*template_path: str*, ***context*) → str

Отрендерить jinja2-шаблон в строку

Параметры

- `template_path` – относительный путь до шаблона, ищется в текущей папке проекта и в папках плагинов
- `context` – контекст шаблона

Глава 6

Базовые команды (cmd)

Модуль carnival.cmd содержит базовые команды для взаимодействия с сервером. Его цель - оставаться простым и помогать в написании шагов (Step).

Для написания сложных сценариев предполагается использовать шаги(Step).

Основные шаги доступны в отдельном репозитории: <https://github.com/carnival-org/carnival-contrib>.

6.1 Cli

`carnival.cmd.cli.is_cmd_exist`(*c: carnival.hosts.base.Connection*, *cmd_ name: str*) → bool

Проверить есть ли команда в $PATH

Параметры

- `c` – Конект с хостом
- `command` – Команда

`carnival.cmd.cli.run`(*c: carnival.hosts.base.Connection*, *command: str*, *warn: bool = True*, *hide: bool = False*, *cwd: Optional[str] = None*) → carnival.hosts.base.Result

Запустить команду

Параметры

- `c` – Конект с хостом
- `command` – Команда для запуска
- `warn` – Вывести stderr
- `hide` – Скрыть вывод команды
- `cwd` – Перейти в папку при выполнении команды

6.2 Filesystem

carnival.cmd.fs.ensure_dir_exists(*c: carnival.hosts.base.Connection*, *path: str*, *user: Optional[str] = None*, *group: Optional[str] = None*, *mode: Optional[str] = None*) → None

Проверить что директория существует и параметры соответствуют заданным

Параметры

- `c` – Конект с хостом
- `path` – путь до директории
- `user` – владелец
- `group` – группа
- `mode` – права

carnival.cmd.fs.is_dir_exists(*c: carnival.hosts.base.Connection*, *dir_path: str*) → bool

Узнать существует ли директория

Параметры

- `c` – Конект с хостом
- `dir_path` – путь до директории

carnival.cmd.fs.is_file_contains(*c: carnival.hosts.base.Connection*, *filename: str*, *text: str*, *exact: bool = False*, *escape: bool = True*) → bool

Содержит ли файл текст

Параметры

- `c` – Конект с хостом
- `filename` – путь до файла
- `text` – текст который нужно искать
- `exact` – точное совпадение
- `escape` – экранировать ли текст

carnival.cmd.fs.is_file_exists(*c: carnival.hosts.base.Connection*, *path: str*) → bool

Проверить существует ли файл

Параметры

- `c` – Конект с хостом
- `path` – путь до файла

carnival.cmd.fs.mkdirs(*c: carnival.hosts.base.Connection*, **dirs*) → List[carnival.hosts.base.Result]

Создать директории

Параметры

- `c` – Конект с хостом
- `dirs` – пути которые нужно создать

6.3 System

carnival.cmd.system.get_current_user_id(*c: carnival.hosts.base.Connection*) → int
Получить id текущего пользователя

Параметры c – Конект с хостом

carnival.cmd.system.get_current_user_name(*c: carnival.hosts.base.Connection*) → str
Получить имя текущего пользователя

Параметры c – Конект с хостом

carnival.cmd.system.is_current_user_root(*c: carnival.hosts.base.Connection*) → bool
Проверить что текущий пользователь - *root*

Параметры c – Конект с хостом

carnival.cmd.system.set_password(*c: carnival.hosts.base.Connection*, *username: str*, *password: str*) → carnival.hosts.base.Result
Установить пароль пользователю

Параметры

- c – Конект с хостом
- username – Пользователь
- password – Новый пароль

6.4 Transfer

carnival.cmd.transfer.get(*c: carnival.hosts.base.Connection*, *remote: str*, *local: str*, *preserve_mode: bool = True*) → None
Скачать файл с сервера <http://docs.fabfile.org/en/2.5/api/transfer.html#fabric.transfer.Transfer.get>

Параметры

- c – Конект с хостом
- remote – путь до файла на сервере
- local – путь куда сохранить файл
- preserve_mode – сохранить права

carnival.cmd.transfer.put(*c: carnival.hosts.base.Connection*, *local: str*, *remote: str*, *preserve_mode: bool = True*) → None
Закачать файл на сервер <http://docs.fabfile.org/en/2.5/api/transfer.html#fabric.transfer.Transfer.put>

Параметры

- c – Конект с хостом
- local – путь до локального файла
- remote – путь куда сохранить на сервере
- preserve_mode – сохранить права

carnival.cmd.transfer.put_template(*c: carnival.hosts.base.Connection*, *template_path: str*, *remote: str*, ***context*) → None

Отрендерить файл с помощью jinja-шаблонов и закачать на сервер См раздел templates.

<http://docs.fabfile.org/en/2.5/api/transfer.html#fabric.transfer.Transfer.put>

Параметры

- `c` – Конект с хостом
- `template_path` – путь до локального файла jinja
- `remote` – путь куда сохранить на сервере
- `context` – контекс для рендеринга jinja2

carnival.cmd.transfer.rsync(*host: carnival.hosts.base.Host*, *source: str*, *target: str*, *rsync_opts: str = '–progress -pthrvz –timeout=60'*, *ssh_opts: str = ''*, *rsync_command: str = 'rsync'*, *hide: bool = False*) → carnival.hosts.base.Result

Залить папку с локального диска на сервер по rsync

Параметры

- `host` – сервер куда заливать
- `source` – локальный путь до папки
- `target` – путь куда нужно залить
- `rsync_opts` – параметры команды rsync
- `ssh_opts` – параметры ssh
- `rsync_command` – путь до rsync
- `hide` – скрыть результаты выполнения

Глава 7

Indices and tables

- genindex
- modindex
- search

Содержание модулей Python

C

Алфавитный указатель

Символы

A

C

E

G

H

I

L

M

N

P

R

S

T

V