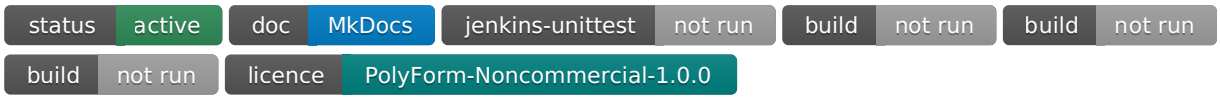
anytraverser

User Manual

chacha

Table of contents



1. Python project template

A nice template to start blank python projets.

This template automate a lot of handy things and allow CI/CD automatic releases generation.

It is also collectings data to feed Jenkins build.

Checkout Latest Documentation.

1.1 Features

1.1.1 Generic pipeline skeleton:

```
- Prepare
- GetCode
- BuildPackage
- Install
- CheckCode
- PlotMetrics
- RunUnitTests
- GenDOC
- PostRelease
```

1.1.2 CI/CD Environment

```
- Jenkins
- Gitea (with patch for dynamic Readme variables: https://chacha.ddns.net/gitea/chacha/GiteaMarkupVariable)
- Docker
- MkDocsWeb
```

1.1.3 CI/CD Helper libs

```
- VirtualEnv
- Changelog generation based on commits
- copier
- pylint + pylint_json2html
- mypy
- unittest + xmlrunner + junitparser + junit2htmlreport
- mkdocs
```

1.1.4 Python project

```
- Full .toml implementation
- .whl automatic generation
- dynamic versionning using git repository
- embedded unit-test
```

2. Usage

2.1 Pulvinar dolor

Donec dapibus est fermentum justo volutpat condimentum. Integer quis nunc neque. Donec dictum vehicula justo, in facilisis ex tincidunt in. Vivamus sollicitudin sem dui, id mollis orci facilisis ut. Proin sed pulvinar dolor. Donec volutpat commodo urna imperdiet pulvinar. Fusce eget aliquam risus. Vivamus viverra luctus ex, in finibus mi. Nullam elementum dapibus mollis. Ut suscipit volutpat ex, quis feugiat lacus consectetur eu.

2.2 Condimentum faucibus

Quisque auctor egestas sem, luctus suscipit ex maximus vitae. Duis facilisis augue et condimentum faucibus. Donec cursus, enim a sagittis egestas, lectus lorem eleifend libero, at tincidunt leo magna at libero. Nunc eros velit, suscipit luctus tempor vel, finibus et est. Curabitur efficitur pretium pulvinar. Donec urna lectus, vulputate quis turpis sed, placerat congue urna. Phasellus aliquet fermentum quam, non auctor elit porta nec. Morbi eu ligula at nisl ultricies condimentum vitae id ante.

2.3 Aliquam lacinia

In volutpat lorem ex, et fringilla nibh faucibus quis. Mauris et arcu elementum, auctor dui vitae, egestas arcu. Duis sit amet aliquam quam. Phasellus a odio turpis. Etiam tristique mi eu enim varius, eget facilisis est vestibulum. Aliquam lacinia nec purus sed luctus. Cras at laoreet erat.

3. Reference

3.1 Anytraverser

3.1.1 metadata

Metadata module module

Attributes

__Name__ `module-attribute`

```
__Name__ = metadata['Name']
```

__Summuary__ `module-attribute`

```
__Summuary__ = metadata['Summary']
```

__version__ `module-attribute`

```
__version__ = version('anytraverser')
```

dist `module-attribute`

```
dist = distribution('anytraverser')
```

3.1.2 Context

Classes

AnnotationWalkerCtx

```
AnnotationWalkerCtx(origin: Any, args: Any, layer: int, parent: Optional[Self] = None, allowed_types: set[type, ...] = frozenset(), allowed_annotations:
dict[str, Any] = {})
```

Attributes

__allowed_annotations `instance-attribute`

```
__allowed_annotations: dict[str, Any] = allowed_annotations
```

__allowed_types `instance-attribute`

```
__allowed_types: set[type, ...] = allowed_types
```

__arg_index `instance-attribute`

```
__arg_index: int | None = None
```

__edge_token `instance-attribute`

```
__edge_token: Any | None = None
```

__ext `instance-attribute`

```
__ext: dict[Any, ChainMap] = {}
```

__layer `instance-attribute`

```
__layer = layer
```

__node_role `instance-attribute`

```
__node_role: NodeRole | None = ELEM
```

__origin `instance-attribute`

```
__origin = origin
```

__parent `instance-attribute`

```
__parent = parent
```

allowed_annotations `property`

```
allowed_annotations: Mapping[str, Any]
```

allowed_types `property`

```
allowed_types: FrozenSet[type]
```

arg_index `property`

```
arg_index: int | None
```

args `instance-attribute`

```
args = args
```

edge_token `property`

```
edge_token: Any | None
```

layer `property`

```
layer: int
```

node_role `property`

```
node_role: NodeRole | None
```

origin `property`

```
origin: Any
```

parent `property`

```
parent: Self
```

Functions

ns

```
ns(owner: Any) -> ChainMap
```

A per-trigger overlay namespace that inherits from parent ctx. Use as: bag = ctx.ns(self); bag['whatever'] = ... Lookups fall back to parent's bag automatically.

NodeRole

Bases: `Enum`

Attributes

ANNO `class-attribute` `instance-attribute`

```
ANNO = auto()
```

ARG `class-attribute` `instance-attribute`

```
ARG = auto()
```

BRANCH `class-attribute` `instance-attribute`

```
BRANCH = auto()
```

ELEM `class-attribute` `instance-attribute`

```
ELEM = auto()
```

KEY `class-attribute` `instance-attribute`

```
KEY = auto()
```

UNION `class-attribute` `instance-attribute`

```
UNION = auto()
```

VAL `class-attribute` `instance-attribute`

```
VAL = auto()
```

3.1.3 Errors

Classes

IncompletelyAnnotatedField

Bases: `RuntimeError`

UnsupportedFieldType

Bases: `RuntimeError`

3.1.4 Protocols

Classes

AnnotationTrigger

Functions

end_trigger

```
end_trigger(ctx: Optional[AnnotationWalkerCtx]) -> None
```

init_trigger

```
init_trigger() -> None
```

process_annotated

```
process_annotated(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_dict

```
process_dict(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_enter

```
process_enter(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_exit

```
process_exit(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_list

```
process_list(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_set

```
process_set(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_simple

```
process_simple(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_tuple

```
process_tuple(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_union

```
process_union(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_unknown

```
process_unknown(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

TriggerResult `dataclass`

```
TriggerResult(replace_with: Any | None = None, skip_children: bool = False, restart_with: Any | None = None)
```

Attributes

replace_with `class-attribute` `instance-attribute`

```
replace_with: Any | None = None
```

restart_with `class-attribute` `instance-attribute`

```
restart_with: Any | None = None
```

skip_children `class-attribute` `instance-attribute`

```
skip_children: bool = False
```

Functions

passthrough `staticmethod`

```
passthrough() -> Self
```

replace `staticmethod`

```
replace(value: Any) -> Self
```

restart `staticmethod`

```
restart(value: Any) -> Self
```

skip `staticmethod`

```
skip() -> Self
```

3.1.5 Data

Errors

3.1.6 Triggers

Test trigger

CLASSES

DataValidation

```
DataValidation(value: Any)
```

Bases: `AnnotationTrigger`

Functions

end_trigger

```
end_trigger(ctx: AnnotationWalkerCtx)
```

init_trigger

```
init_trigger() -> None
```

process_annotated

```
process_annotated(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_dict

```
process_dict(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_enter

```
process_enter(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_exit

```
process_exit(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_list

```
process_list(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_set

```
process_set(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_simple

```
process_simple(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_tuple

```
process_tuple(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_union

```
process_union(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_unknown

```
process_unknown(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

LAMSchemaValidation

Bases: `AnnotationTrigger`

Functions

end_trigger

```
end_trigger(ctx: AnnotationWalkerCtx)
```

init_trigger

```
init_trigger() -> None
```

process_allowed

```
process_allowed(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_annotated

```
process_annotated(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_dict

```
process_dict(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_exit

```
process_exit(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_list

```
process_list(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_set

```
process_set(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_tuple

```
process_tuple(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

process_union

```
process_union(ctx: AnnotationWalkerCtx) -> None | TriggerResult
```

Validation trigger

CLASSES

HorizontalValidationTrigger

```
HorizontalValidationTrigger(value: Any, *, strict_bool: bool = False, collect_all: bool = True, union_summary_max_per_branch: int = 3, show_branch_types: boo
l = True)
```

Bases: `AnnotationTrigger`

Functions

end_trigger

```
end_trigger(ctx: Optional[AnnotationWalkerCtx])
```

process_allowed

```
process_allowed(ctx: AnnotationWalkerCtx)
```

process_annotated

```
process_annotated(ctx: AnnotationWalkerCtx)
```

process_dict

```
process_dict(ctx: AnnotationWalkerCtx)
```

process_exit

```
process_exit(ctx: AnnotationWalkerCtx)
```

process_list

```
process_list(ctx: AnnotationWalkerCtx)
```

process_set

```
process_set(ctx: AnnotationWalkerCtx)
```

process_tuple

```
process_tuple(ctx: AnnotationWalkerCtx)
```

process_union

```
process_union(ctx: AnnotationWalkerCtx)
```

SchemaValidationError

```
SchemaValidationError(path: tuple[Any, ...], msg: str)
```

Bases: `TypeError`

Attributes

path `instance-attribute`

```
path = path
```

Functions

3.1.7 Walkers

Annotation walker

CLASSES

AnnotationWalker

```
AnnotationWalker(ann: Any, triggers: tuple[AnnotationTrigger, ...], **kwargs)
```

Attributes

DEFAULT_ALLOWED_ANNOTATIONS `class-attribute` `instance-attribute`

```
DEFAULT_ALLOWED_ANNOTATIONS: dict[str, Any] = frozendict({'Union': Union, 'Optional': Optional, 'List': List, 'Dict': Dict, 'Tuple': Tuple, 'Set': Set, 'Annotated': Annotated, 'int': int, 'str': str, 'float': float, 'bool': bool, 'complex': complex, 'bytes': bytes, 'None': type(None), 'list': list, 'dict': dict, 'set': set, 'tuple': tuple})
```

DEFAULT_ALLOWED_TYPES `class-attribute` `instance-attribute`

```
DEFAULT_ALLOWED_TYPES = frozenset({str, int, float, complex, bool, bytes, NoneType})
```

__ann `instance-attribute`

```
__ann = ann
```

Functions

run

```
run() -> TriggerResult
```