TODO zz a

Release TODO zz a

unknown

May 31, 2021

CONTENTS

TODO Anise is a Python-based execution engine for automation tasks.Automation tasks exist in software development, and probably all kinds of other sectors. They typicallyrequire the execution of different smaller and larger tools.Complex tasks often need a sequence of many steps to execute, with some steps having dependenciesto each other.Manually triggering all these steps in the graphical interfaces of all the involved tools ispossible in theory, but will generate errors and frustration after some cycles.The automation interfaces of those tools are sometimes easier, but sometimesthey are error-prone. Some tasks may also need to ask the user for some information in an interactive way.Some smaller parts might also be machine-specific (e.g. filesystem paths or the code how toaccess a password vault), while the entire task must be runnable on some different machines.In some situations, this can lead to a rather intransparent forest of different tools, with uniqueoddnesses and special conventions. As the number of different project increases, you will see more and moredifferent tools, often doing a similar job, but for different platforms or frameworks and, of course,with different usage conventions. Spontaneously written glue scripts help in the beginning, butwill explode as the complexity exceeds some threshold.Typical tasks in software development could be:

- Generating documentation- Testing- Automatic code generation- Creating packages- Creating a homepage, automatically built from the available version information, the packages, the documentation and so on- Deploying this homepage to a web server- Handling version information - e.g. print it in the manual- and many moreThe Anise framework allows you to implement all those tasks in a structured but generic way in a combination of XML andPython code. Once you have created this stuff at a defined place in your project, Anise lets you easily execute yourtasks from command line (or from any editor if you embed it somehow). This gives you a common and easy interfaceto all your ‘tool glue’ code.The Anise engine executes arbitrary Python sourcecode and provides some additional services like logging, parameter passing from command line, basic graphical userinterface support, a plugin interface, a flexible event system, injecting code and data from other place,dependencies between code fragments, and more.On top of this engine, Anise comes with a bunch of implementations that fulfill tasks (or parts of them) ofsoftware development. There is a testing module, a documentation- and homepage-generator, some package buildingmethods and a lot more. The implementations use the event system in many places in order to allow customization ina somewhat technical but very flexible way. Even so, those implementations are rather specific and it depends onthe particular case, if, and how many of those implementations are useful.

CHAPTER ONE

LICENSE

Annize is distributed under the terms of the AGPL 3 license. This also affects all included files without a license header (non-source files like images), unless they are explicitly mentioned as third-party content. Read the ‘Dependencies’ section for included third-party stuff.

CHAPTER TWO

UP-TO-DATE?

Are you currently reading from another source than the homepage? If you are in doubt whether your package is up-to-date, you should visit the project homepage and check that. You are currently reading the documentation for version 5.0.6379.

CHAPTER

THREE

DEPENDENCIES

Annize makes use of some third-party parts.

Required: **Python 3.7**

Required: **GTK 3.24**

Recommended: **GNU/Linux**

Included: **banner image** (License: http://creativecommons.org/licenses/by-sa/3.0; from here)

Included: **font 'Inconsolata'** (for websites; by Raph Levien; License: OFL; from here)

Included: **font 'Khula'** (for websites; by Erin McLaughlin; License: OFL; from here)

Included: **font 'Symbola'** (for logo symbol; License: free for use; from here)

Included: **icon set 'Oxygen'** (some icons; from here)

Included: **third-party project logos** (from here)

CHAPTER

FOUR

USER MANUAL

4.1 Installation

TODO zz

4.2 First Steps

TODO zz TODO zz cli TODO zz file format TODO zz gui

4.3 Project Configuration

TODO zz TODO zz classes TODO zz features/xml namespaces TODO zz names, references and append_to

4.3.1 Model

TODO zz

4.3.2 File Format

TODO zz

4.4 Graphical User Interface

TODO zz

4.5 The Annize Programming Interface

TODO zz TODO zz i18n TODO zz fs

4.5.1 Features

TODO zz

CHAPTER

FIVE

COMMAND LINE INTERFACE REFERENCE

```
usage: annize [-h] [--project PROJECT] {do} ...
```

5.1 Positional Arguments

command

Possible choices: do

The command to execute.

5.2 Named Arguments

--project

Project directory or Annize configuration file (otherwise: the current working directory). Annize will automatically try to find it in parent directories as well.

5.3 Sub-commands:

5.3.1 do

Execute a task.

```
annize do [-h] [--with-answers-from-json-file WITH_ANSWERS_FROM_JSON_FILE]
          [--with-answers-from-json-string WITH_ANSWERS_FROM_JSON_STRING]
          [--with-answer WITH_ANSWER WITH_ANSWER]
          [taskname]
```

Positional Arguments

taskname

Task name.

Default: “taskchooser”

Named Arguments

--with-answers-from-json-file an_Cli_withAnswersFromJsonFile

Default: []

--with-answers-from-json-string an_Cli_withAnswersFromJsonString

Default: []

--with-answer an_Cli_withAnswer

Default: []

CHAPTER

SIX

API REFERENCE

6.1 annize namespace

6.1.1 Subpackages

annize.aux package

Submodules

annize.aux._project_infos module

annize.callconv package

exception annize.callconv.**MultipleValuesForSingleArgumentError**(*argname*)

Bases: TypeError

Parameters **argname** (*str*) –

class annize.callconv.**ParametersFromChildNodesInfos**(*parameter_name: str, multiple: Union[bool, NoneType], if_passing: Union[Callable[[Any], bool], NoneType], if_typehint_matches: bool*)

Bases: object

Parameters

- **parameter_name** (*str*) –
- **multiple** (*Optional[bool]*) –
- **if_passing** (*Optional[Callable[[Any], bool]]*) –
- **if_typehint_matches** (*bool*) –

Return type None

if_passing: Optional[Callable[[Any], bool]]

if_typehint_matches: bool

multiple: Optional[bool]

parameter_name: str

class annize.callconv.**_CallHelper**

Bases: object

static _CallHelper__convert_kwargs_from_string(*argumentinfos*, *args*, *kwargs*)

static _CallHelper__shift_args_to_kwargs(*calltype*, *argumentinfos*, *args*, *kwargs*)

classmethod call(*calltype*, *args*, *kwargs*)

Parameters

- **args** (*Iterable*) –
- **kwargs** (*Dict*) –

annize.callconv.**always_true**(*_*)

Parameters _ (*Any*) –

Return type bool

annize.callconv.**call**(*calltype*, *args*, *kwargs*)

Parameters

- **args** (*Iterable*) –
- **kwargs** (*Dict*) –

annize.callconv.**get_parameters_from_childnodes_infos**(*calltype*)

Parameters **calltype** (*Type*) –

Return type List[*annize.callconv.ParametersFromChildNodesInfos*]

annize.callconv.**parameter_from_childnodes**(*paramname*, *multiple=None*, *if_passing=None*, *if_typehint_matches=True*)

Parameters

- **paramname** (*str*) –
- **multiple** (*Optional[bool]*) –
- **if_passing** (*Optional[Callable[[Any], bool]]*) –
- **if_typehint_matches** (*bool*) –

Submodules

annize.callconv.typehints module

class annize.callconv.typehints.**ListTypeInfo**(*name*, *innertypeinfo*)

Bases: *annize.callconv.typehints.TypeInfo*

Parameters **name** (*str*) –

property inner

class annize.callconv.typehints.**TypeInfo**(*name*, *resolvedtype*, *constructfromstringfct*)

Bases: object

Parameters

- **name** (*str*) –
- **resolvedtype** (*Type*) –

- **constructfromstringfct** (*Optional[Callable[[str], Any]]*) –

construct_from_string (*strvalue*)

Parameters **strvalue** (*str*) –

Return type Any

property is_constructable_from_string

property name

property resolved_type

annize.callconv.typehints.**get_arguments_infos** (**, for_callable*)

Parameters **for_callable** (*Callable*) –

Return type Dict[str, *annize.callconv.typehints.TypeInfo*]

annize.callconv.typehints.**get_typeinfo** (*for_type*)

Return type *annize.callconv.typehints.TypeInfo*

annize.context package

class annize.context.**Context**

Bases: object

_METADATA_IS_TOPLEVEL_OBJECT = 'annize..is_toplevel_object'

__object_metadata_dict (*obj*)

Parameters **obj** (*Any*) –

Return type Dict[str, Optional[Any]]

__object_raw_name (*obj*)

Parameters **obj** (*Any*) –

Return type str

__put_object (*name, obj*)

Parameters

- **name** (*str*) –
- **obj** (*Any*) –

Return type None

add_object (*obj*)

Parameters **obj** (*Any*) –

Return type str

get_object_metadata (*obj, key, defaultvalue=None*)

Parameters

- **obj** (*Any*) –
- **key** (*str*) –
- **defaultvalue** (*Optional[Any]*) –

Return type Optional[Any]

`is_friendly_name`(*name*)

Parameters **`name`** (*`str`*) –

Return type bool

`is_toplevel_object`(*obj*)

Parameters **`obj`** (*`Any`*) –

Return type bool

`mark_object_as_toplevel`(*obj*)

Parameters **`obj`** (*`Any`*) –

Return type None

`object_by_name`(*name*, *defaultvalue=None*, ***, *create_nonexistent=False*)

Parameters

- **`name`** (*`str`*) –
- **`defaultvalue`** (*`Optional[Any]`*) –
- **`create_nonexistent`** (*`bool`*) –

Return type Optional[Any]

`object_name`(*obj*)

Parameters **`obj`** (*`Any`*) –

Return type str

`object_names`(*obj*)

Parameters **`obj`** (*`Any`*) –

Return type List[str]

`objects_by_type`(*objtype*, *toplevel_only=True*)

Parameters

- **`objtype`** (*`Type[_T]`*) –
- **`toplevel_only`** (*`bool`*) –

Return type List[_T]

`set_object_metadata`(*obj*, *key*, *value=None*)

Parameters

- **`obj`** (*`Any`*) –
- **`key`** (*`str`*) –
- **`value`** (*`Optional[Any]`*) –

Return type None

`set_object_name`(*obj*, *name*)

Parameters

- **`obj`** (*`Any`*) –

- **name** (*str*) –

Return type None

exception annize.context.**OutOfContextError**

Bases: TypeError

annize.context.**add_object**(*obj*)

Parameters **obj** (*Any*) –

Return type str

annize.context.**current**()

Return type *annize.context.Context*

annize.context.**get_object_metadata**(*obj*, *key*, *defaultvalue=None*)

Parameters

- **obj** (*Any*) –
- **key** (*str*) –
- **defaultvalue** (*Optional[Any]*) –

Return type Optional[Any]

annize.context.**is_friendly_name**(*name*)

Parameters **name** (*str*) –

Return type bool

annize.context.**is_toplevel_object**(*obj*)

Parameters **obj** (*Any*) –

Return type bool

annize.context.**object_by_name**(*name*, *defaultvalue=None*, ***, *create_nonexistent=False*)

Parameters

- **name** (*str*) –
- **defaultvalue** (*Optional[Any]*) –
- **create_nonexistent** (*bool*) –

Return type Optional[Any]

annize.context.**object_name**(*obj*)

Parameters **obj** (*Any*) –

Return type str

annize.context.**object_names**(*obj*)

Parameters **obj** (*Any*) –

Return type List[str]

annize.context.**objects_by_type**(*objtype*, *toplevel_only=True*)

Parameters

- **objtype** (*Type[_T]*) –

- **toplevel_only** (*bool*) –

Return type List[_T]

annize.context.**set_object_metadata**(*obj*, *key*, *value=None*)

Parameters

- **obj** (*Any*) –
- **key** (*str*) –
- **value** (*Optional[Any]*) –

Return type None

annize.context.**set_object_name**(*obj*, *name*)

Parameters

- **obj** (*Any*) –
- **name** (*str*) –

Return type None

annize.data package

Submodules

annize.data.color module

class annize.data.color.**Color**(**, *red*, *green*, *blue*)

Bases: object

Parameters

- **red** (*float*) –
- **green** (*float*) –
- **blue** (*float*) –

property blue

property green

property html_color_spec

property red

scalehue(*, *brightness*, *saturation=None*)

Parameters

- **brightness** (*float*) –
- **saturation** (*Optional[float]*) –

Return type *annize.data.color.Color*

transformed(*, *brightness=None*, *saturation=None*)

Parameters

- **brightness** (*Optional[float]*) –

- **saturation** (*Optional[float]*) –

Return type *annize.data.color.Color*

annize.data.container module

class annize.data.container.**Basket**(**a*, ***b*)

Bases: `list`

annize.data.unique module

class annize.data.unique.**UniqueId**(*localid=None*)

Bases: `object`

Parameters **localid** (*Optional[str]*) –

__long_str(*txt*)

Parameters **txt** (*str*) –

Return type str

__uniqueid_counter = 0

__uniqueid_lock = <unlocked _thread.lock object>

property long_str

property processonly_long_str

property processonly_short_str

property short_str

annize.data.version module

class annize.data.version.**AbstractVersionPatternSegment**

Bases: `object`

_name_anon(*namep*)

Parameters **namep** (*str*) –

Return type None

_str_to_val(*txt*)

Parameters **txt** (*str*) –

Return type object

regexp_string()

Return type str

property segment_names

segments_tuples_to_text(*segments_tuples*)

Parameters **segments_tuples** (TODO) –

Return type str

class annize.data.version.**ConcatenatedVersionPatternSegment**(**, segments*)

Bases: *annize.data.version.AbstractVersionPatternSegment*

Parameters **segments** (*Iterable[*annize.data.version.AbstractVersionPatternSegment*]*) –

_name_anon(*namep*)

regexp_string()

Return type str

property segment_names

segments_tuples_to_text(*segments_tuples*)

Parameters **segments_tuples** (TODO) –

Return type str

class annize.data.version.**NumericVersionPatternSegment**(**, partname=None*)

Bases: *annize.data.version.AbstractVersionPatternSegment*

Parameters **partname** (*Optional[str]*) –

_name_anon(*namep*)

_str_to_val(*txt*)

Parameters **txt** (*str*) –

Return type object

property partname

regexp_string()

Return type str

property segment_names

segments_tuples_to_text(*segments_tuples*)

Parameters **segments_tuples** (TODO) –

Return type str

class annize.data.version.**OptionalVersionPatternSegment**(**, segments*)

Bases: *annize.data.version.AbstractVersionPatternSegment*

Parameters **segments** (*Iterable[*annize.data.version.AbstractVersionPatternSegment*]*) –

_name_anon(*namep*)

regexp_string()

Return type str

property segment_names

segments_tuples_to_text(*segments_tuples*)

Parameters **segments_tuples** (TODO) –

Return type str

class annize.data.version.**SeparatorVersionPatternSegment**(**, text*)

Bases: *annize.data.version.AbstractVersionPatternSegment*

Parameters **text** (*str*) –

regexp_string()

Return type str

segments_tuples_to_text(*segments_tuples*)

Parameters **segments_tuples** (TODO) –

Return type str

class annize.data.version.**Version**(**, text=None, pattern=None, **segment_values*)

Bases: object

Parameters

- **text** (*str*) –
- **pattern** (annize.data.version.VersionPattern) –

property pattern

property segments_tuples

property segments_values

property text

class annize.data.version.**VersionPattern**(**, segments*)

Bases: object

Parameters **segments** (*Iterable[*annize.data.version.AbstractVersionPatternSegment*]*) –

property segment_names

property segments

segments_tuples_to_text(*segments_tuples*)

Parameters **segments_tuples** (TODO) –

Return type str

text_to_segments_tuples(*text*)

Parameters **text** (*str*) –

Return type *TODO*

annize.features namespace

Subpackages

annize.features.changelog namespace

Submodules

annize.features.changelog.common module

class annize.features.changelog.common.**ByVersionControlSystemCommitMessagesChangelog**

Bases: *annize.features.changelog.common.Changelog*

_S_CHANGE = '##CHANGE:'

_S_LABEL = '##LABEL:'

property entries

class annize.features.changelog.common.**Changelog**(*, *entries=()*)

Bases: object

Parameters **entries** (*Iterable[annize.features.changelog.common.Entry]*) –

property entries

class annize.features.changelog.common.**Entry**(*, *version=None*, *time=None*, *items*)

Bases: object

Parameters

- **version** (annize.data.version.Version) –
- **time** (*datetime.datetime*) –
- **items** (*Iterable[annize.features.changelog.common.Item]*) –

property items

property time

property version

class annize.features.changelog.common.**Item**(*, *text*)

Bases: object

Parameters **text** (annize.i18n.TrStr) –

property text

annize.features.changelog.common.**_items_matcher**(*o*)

annize.features.changelog.common.**default_changelog**()

Return type *annize.features.changelog.common.Changelog*

annize.features.datainjections namespace

Submodules

annize.features.datainjections.common module

class annize.features.datainjections.common.**Inject**(*, *injection*, *destination*)

Bases: object

Parameters

- **injection** (annize.features.datainjections.common.Injection) –
- **destination** (annize.fs.FsEntry) –

class annize.features.datainjections.common.**Injection**

Bases: *annize.fs.AdHocFsEntry*

_abc_impl = <_abc_data object>

_initialize()

inject (*destination*)

Parameters **destination** (annize.fs.FsEntry) –

annize.features.datainjections.python module

class annize.features.datainjections.python.**Injection** (*, *filename='_project_infos.py'*, *version=None*)

Bases: *annize.features.datainjections.common.Injection*

Parameters

- **filename** (*str*) –
- **version** (annize.data.version.Version) –

_abc_impl = <_abc_data object>

inject (*destination*)

Parameters **destination** (annize.fs.FsEntry) –

annize.features.dependencies namespace

Submodules

annize.features.dependencies.common module

class annize.features.dependencies.common.**Artwork** (*, *kind=None*, *label*, *origin=None*, *comment=None*, *license=None*)

Bases: *annize.features.dependencies.common.Dependency*

Parameters

- **origin** (*str*) –
- **comment** (*str*) –
- **license** (*str*) –

class annize.features.dependencies.common.**Dependency** (*, *kind=None*, *label=None*, *comment=None*, *icon=None*, *importance=0*)

Bases: object

Parameters

- **kind** (annize.features.dependencies.common.Kind) –
- **label** (*str*) –
- **comment** (*str*) –
- **icon** (*str*) –
- **importance** (*int*) –

property comment

property icon

property importance

property kind

property label

class annize.features.dependencies.common.**GnuLinux**(*, *kind=None*, *comment=''*)
Bases: *annize.features.dependencies.common.Dependency*

class annize.features.dependencies.common.**Included**(***b*)
Bases: *annize.features.dependencies.common.Kind*

class annize.features.dependencies.common.**Kind**(*, *label*, *importance*, *flags=None*)
Bases: object

Parameters

- **label** (*str*) –
- **importance** (*int*) –
- **flags** (*List[object]*) –

property flags

property importance

property label

class annize.features.dependencies.common.**Recommended**(***b*)
Bases: *annize.features.dependencies.common.Kind*

class annize.features.dependencies.common.**Required**(***b*)
Bases: *annize.features.dependencies.common.Kind*

annize.features.dependencies.common.**dependencies_to_rst_text**(*dependencies*)

Parameters dependencies (*List[*annize.features.dependencies.common.Dependency*]*) –

Return type str

annize.features.dependencies.python module

class annize.features.dependencies.python.**Python**(*, *version*, *kind=None*)
Bases: *annize.features.dependencies.common.Dependency*

Parameters

- **version** (*str*) –
- **kind** (annize.features.dependencies.common.Kind) –

annize.features.distributables namespace

Subpackages

annize.features.distributables.store namespace

Submodules

annize.features.distributables.store.pypi module

class annize.features.distributables.store.pypi.**Connection**(**, token*)

Bases: object

Parameters **token** (*str*) –

property token

class annize.features.distributables.store.pypi.**Upload**(**, source, connection*)

Bases: object

Parameters

- **source** (annize.fs.FsEntry) –
- **connection** (annize.features.distributables.store.pypi.Connection) –

Submodules

annize.features.distributables.common module

class annize.features.distributables.common.**File**

Bases: *annize.fs.FsEntry*

_abc_impl = <_abc_data object>

class annize.features.distributables.common.**Group**(**, title, files, description='', packagestore=None*)

Bases: object

Parameters

- **title** (*str*) –
- **files** (*List[*annize.fs.FsEntry*]*) –
- **description** (*str*) –
- **packagestore** (annize.features.distributables.common.PackageStore) –

__get_files_from_packagestore()

Return type Iterable[*annize.fs.FsEntry*]

__getname()

Return type str

__store_files_to_packagestore()

Return type None

property description

get_files()

Return type List[*annize.fs.FsEntry*]

property packagestore

property title

class annize.features.distributables.common.**PackageStore**

Bases: object

get_package(*, *name*)

Parameters **name** (*str*) –

Return type Optional[Iterable[*annize.fs.FsEntry*]]

get_package_history(*, *name*, *limit=3*)

Parameters

- **name** (*str*) –
- **limit** (*int*) –

Return type Iterable[Iterable[*annize.fs.FsEntry*]]

store_package(*fsentries*, *, *name*)

Parameters

- **fsentries** (*Iterable[annize.fs.FsEntry]*) –
- **name** (*str*) –

Return type None

annize.features.distributables.debian module

annize.features.distributables.debian.**AdministrationUtilitiesSection**

alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**ApplicationsAccessibilityCategory**

alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsAmateurradioCategory**

alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsDatamanagementCategory**

alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsEditorsCategory**

alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsEducationCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsEmulatorsCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsFilemanagementCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsGraphicsCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsMobiledevicesCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsNetworkCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsNetworkCommunicationCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsNetworkFiletransferCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsNetworkMonitoringCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsNetworkWebbrowsingCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsNetworkWebnewsCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsOfficeCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsProgrammingCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsProjectmanagementCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsScienceAstronomyCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsScienceBiologyCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsScienceCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsScienceChemistryCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsScienceDataanalysisCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsScienceElectronicsCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsScienceEngineeringCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsScienceGeoscienceCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsScienceMathematicsCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsScienceMedicineCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsSciencePhysicsCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsScienceSocialCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsShellsCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsSoundsCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsSystemAdministrationCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsSystemCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsSystemHardwareCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsSystemLanguageenvironmentCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsSystemMonitoringCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsSystemPackagemanagementCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsSystemSecurityCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsTerminalemulatorsCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsTextCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsTvandradioCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsVideoCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsViewersCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ApplicationsWebdevelopmentCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

class annize.features.distributables.debian.**Category**(*, *debian_name*, *freedesktop_name*)
Bases: object

Parameters

- **debian_name** (*str*) –
- **freedesktop_name** (*str*) –

property debian_name

property freedesktop_name

annize.features.distributables.debian.**CommunicationProgramsSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**DatabasesSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**DebianInstallerUdebPackagesSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**DebugPackagesSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**DevelopmentSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**DocumentationSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**EditorsSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**EducationSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**ElectronicsSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**EmbeddedSoftwareSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

class annize.features.distributables.debian.**ExecutableLink**(*, *path*, *name=None*)
Bases: object

Parameters

- **path** (*str*) –
- **name** (*str*) –

property name

property path

annize.features.distributables.debian.**FontsSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**GamesActionCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**GamesAdventureCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**GamesBlocksCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**GamesBoardCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**GamesCardCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**GamesPuzzlesCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**GamesSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**GamesSimulationCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**GamesStrategyCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**GamesToolsCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**GamesToysCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**GnomeSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**GnuRSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**GnustepSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**GraphicsSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**HamRadioSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**HaskellSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**HelpCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**InterpretersSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**IntrospectionSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**JavaSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**JavascriptSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**KdeSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**KernelsSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**LanguagePacksSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**LibrariesSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**LibraryDevelopmentSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**LispSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**MailSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**MathematicsSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

class annize.features.distributables.debian.**MenuEntry**(**, name, title, category, command, is_gui, icon=None*)
Bases: object

Parameters

- **name** (*str*) –
- **title** (*str*) –
- **category** (annize.features.distributables.debian.Category) –

- **command** (*str*) –
- **is_gui** (*bool*) –
- **icon** (annize.fs.FsEntry) –

property category

property command

property icon

property is_gui

property name

property title

annize.features.distributables.debian.**MetaPackagesSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**MiscellaneousSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**MonoCliSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**NetworkSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**NewsgroupsSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**OcamlSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**OldLibrariesSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**OtherOSsAndFSsSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

class annize.features.distributables.debian.**Package**(**, source, menuentries, executable_links, packagename=None, description=None, summary=None, section=None, homepage_url=None, version=None, documentation=None, author=None, prerm='', postinst='', architecture='all'*)

Bases: *annize.fs.AdHocFsEntry*

Parameters

- **source** (annize.fs.FsEntry) –

- **menuentries** (*List[*annize.features.distributables.debian.MenuEntry*]*) –
- **executable_links** (*List[*annize.features.distributables.debian.ExecutableLink*]*) –
- **packagename** (*Optional[str]*) –
- **description** (*Optional[str]*) –
- **summary** (*Optional[str]*) –
- **section** (annize.features.distributables.debian.Section) –
- **homepage_url** (*Optional[str]*) –
- **version** (annize.data.version.Version) –
- **documentation** (annize.fs.FsEntry) –
- **author** (annize.features.authors.Author) –
- **prerm** (*str*) –
- **postinst** (*str*) –
- **architecture** (*str*) –

class _BuildInfo(*source: annize.fs.FsEntry*, *executable_links: List[annize.features.distributables.debian.ExecutableLink]*, *menuentries: List[annize.features.distributables.debian.MenuEntry]*, *services: List[annize.features.distributables.debian.ServiceDescription]*, *description: str*, *name: str*, *version: annize.data.version.Version*, *homepage: str*, *author: annize.features.authors.Author*, *licensename: str*, *section: Union[annize.features.distributables.debian.Section, NoneType]*, *summary: str*, *documentation_source: Union[annize.fs.FsEntry, NoneType]*, *prerm: str*, *postinst: str*, *architecture: str*, *authorstring: str = None*, *pkgrootpath: str = None*, *pkgpath_debian: str = None*, *pkgpath_documentation: str = None*, *pkgpath_pixmaps: str = None*, *pkgpath_usrbin: str = None*, *config_files: List[str] = None*, *pkgsize: int = None*, *result: annize.fs.FsEntry = None*)

Bases: `object`

_BuildInfo(source: annize.fs.FsEntry, executable_links: List[annize.features.distributables.debian.ExecutableLink], menuentries: List[annize.features.distributables.debian.MenuEntry], services: List[annize.features.distributables.debian.ServiceDescription], description: str, name: str, version: annize.data.version.Version, homepage: str, author: annize.features.authors.Author, licensename: str, section: Union[annize.features.distributables.debian.Section, NoneType], summary: str, documentation_source: Union[annize.fs.FsEntry, NoneType], prerm: str, postinst: str, architecture: str, authorstring: str = None, pkgrootpath: str = None, pkgpath_debian: str = None, pkgpath_documentation: str = None, pkgpath_pixmaps: str = None, pkgpath_usrbin: str = None, config_files: List[str] = None, pkgsize: int = None, result: annize.fs.FsEntry = None)

Parameters

- **source** (annize.fs.FsEntry) –
- **executable_links** (*List[*annize.features.distributables.debian.ExecutableLink*]*) –
- **menuentries** (*List[*annize.features.distributables.debian.MenuEntry*]*) –
- **services** (*List[*annize.features.distributables.debian.ServiceDescription*]*) –

- **description** (*str*) –
- **name** (*str*) –
- **version** (annize.data.version.Version) –
- **homepage** (*str*) –
- **author** (annize.features.authors.Author) –
- **licensename** (*str*) –
- **section** (*Optional[*annize.features.distributables.debian.Section*]*) –
- **summary** (*str*) –
- **documentation_source** (*Optional[*annize.fs.FsEntry*]*) –
- **prerm** (*str*) –
- **postinst** (*str*) –
- **architecture** (*str*) –
- **authorstring** (*str*) –
- **pkgrootpath** (*str*) –
- **pkgpath_debian** (*str*) –
- **pkgpath_documentation** (*str*) –
- **pkgpath_pixmaps** (*str*) –
- **pkgpath_usrbin** (*str*) –
- **config_files** (*List[str]*) –
- **pkgsize** (*int*) –
- **result** (annize.fs.FsEntry) –

Return type None

architecture: str

author: ***annize.features.authors.Author***

authorstring: str = None

config_files: List[str] = None

description: str

documentation_source: Optional[*annize.fs.FsEntry*]

executable_links: List[*annize.features.distributables.debian.ExecutableLink*]

homepage: str

licensename: str

menuentries: List[*annize.features.distributables.debian.MenuEntry*]

name: str

pkgpath_debian: str = None

pkgpath_documentation: str = None

pkgpath_pixmaps: str = None

pkgpath_usrbin: str = None

pkgrootpath: str = None

pkgsize: int = None

postinst: str

prerm: str

result: *annize.fs.FsEntry* **= None**

section: Optional[*annize.features.distributables.debian.Section***]**

services: List[*annize.features.distributables.debian.ServiceDescription***]**

source: *annize.fs.FsEntry*

summary: str

version: *annize.data.version.Version*

_abc_impl = <_abc_data object>

_initialize()

classmethod _mkpackage(*build*)

Parameters build (annize.features.distributables.debian.Package._BuildInfo) –

Return type *annize.fs.FsEntry*

classmethod _mkpackage_correctbuildsourcepermissions(*build*)

Parameters build (annize.features.distributables.debian.Package._BuildInfo) –

Return type None

classmethod _mkpackage_determinesize(*build*)

Parameters build (annize.features.distributables.debian.Package._BuildInfo) –

Return type None

classmethod _mkpackage_dpkg(*build*)

Parameters build (annize.features.distributables.debian.Package._BuildInfo) –

Return type None

classmethod _mkpackage_mkchangelog(*build*)

Parameters build (annize.features.distributables.debian.Package._BuildInfo) –

Return type None

classmethod _mkpackage_mkcopyright(*build*)

Parameters build (annize.features.distributables.debian.Package._BuildInfo) –

Return type None

classmethod _mkpackage_mkdebianconffilesfile(*build*)

Parameters **build** (annize.features.distributables.debian.Package._BuildInfo) –

Return type None

classmethod _mkpackage_mkdebiancontrolfile(*build*)

Parameters **build** (annize.features.distributables.debian.Package._BuildInfo) –

Return type None

classmethod _mkpackage_mkexeclinks(*build*)

Parameters **build** (annize.features.distributables.debian.Package._BuildInfo) –

Return type None

classmethod _mkpackage_mkmenuentries(*build*)

Parameters **build** (annize.features.distributables.debian.Package._BuildInfo) –

Return type None

classmethod _mkpackage_mkprepostcmds(*build*)

Parameters **build** (annize.features.distributables.debian.Package._BuildInfo) –

Return type None

classmethod _mkpackage_mkservices(*build*)

Parameters **build** (annize.features.distributables.debian.Package._BuildInfo) –

Return type None

classmethod _mkpackage_prepareinfos(*build*, *tmpdir*)

Parameters

- **build** (annize.features.distributables.debian.Package._BuildInfo) –
- **tmpdir** (*str*) –

Return type None

annize.features.distributables.debian.**PerlSection**

alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**PhpSection**

alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**PythonSection**

alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**RubySection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**RustSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**ScienceSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**ScreenLockingCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

annize.features.distributables.debian.**ScreenSavingCategory**
alias of annize.features.distributables.debian._debian_category.<locals>.ACategory

class annize.features.distributables.debian.**Section**(*, *name*)
Bases: object

property name

class annize.features.distributables.debian.**ServiceDescription**(*name*, *command*)
Bases: object

Description for Debian services to be included in a package.

Parameters

- **name** (*str*) – The display name.
- **command** (*str*) – The command to be executed.

annize.features.distributables.debian.**ShellsSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**SoundSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**TasksSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**TexSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**TextProcessingSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**UtilitiesSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**VersionControlSystemsSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**VideoSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**VirtualPackagesSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**WebServersSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**WebSoftwareSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**XWindowSystemSoftwareSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**XfceSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**ZopePloneFrameworkSection**
alias of annize.features.distributables.debian._debian_section.<locals>.ASection

annize.features.distributables.debian.**_debian_category**(*debian_name*, *freedesktop_name*)

Parameters

- **debian_name** (*str*) –
- **freedesktop_name** (*str*) –

Return type Type[*annize.features.distributables.debian.Category*]

annize.features.distributables.debian.**_debian_section**(*name*)

Parameters **name** (*str*) –

Return type Type[*annize.features.distributables.debian.Section*]

annize.features.distributables.flatpak module

class annize.features.distributables.flatpak.**EnvironmentVariable**(***b*)
Bases: object

class annize.features.distributables.flatpak.**Filesystem**(***b*)
Bases: object

class annize.features.distributables.flatpak.**FlatpakImage**(***, *source*, *package_name*, *sockets='x11'*, *filesystems='home'*, *shares='network'*)

Bases: `annize.fs.AdHocFsEntry`

Parameters

- **source** (annize.fs.FsEntry) –
- **package_name** (*str*) –
- **sockets** (*Iterable[str]*) –
- **filesystems** (*Iterable[str]*) –
- **shares** (*Iterable[str]*) –

class _BuildInfo(*source: annize.fs.FsEntry, name: str, sdk: str, platform: str, kitversion: Union[str, NoneType], command: Union[str, NoneType], sockets: Iterable[str], filesystems: Iterable[str], shares: Iterable[str], menu_entries: Iterable[annize.features.distributables.flatpak.MenuEntry], environment: Dict[str, str], pkgrootpath: str = None, pkgpath_share: str = None, pkgpath_share_applications: str = None, pkgpath_share_icons: str = None, result: annize.fs.FsEntry = None*)

Bases: `object`

_BuildInfo(source: annize.fs.FsEntry, name: str, sdk: str, platform: str, kitversion: Union[str, NoneType], command: Union[str, NoneType], sockets: Iterable[str], filesystems: Iterable[str], shares: Iterable[str], menu_entries: Iterable[annize.features.distributables.flatpak.MenuEntry], environment: Dict[str, str], pkgrootpath: str = None, pkgpath_share: str = None, pkgpath_share_applications: str = None, pkgpath_share_icons: str = None, result: annize.fs.FsEntry = None)

Parameters

- **source** (annize.fs.FsEntry) –
- **name** (*str*) –
- **sdk** (*str*) –
- **platform** (*str*) –
- **kitversion** (*Optional[str]*) –
- **command** (*Optional[str]*) –
- **sockets** (*Iterable[str]*) –
- **filesystems** (*Iterable[str]*) –
- **shares** (*Iterable[str]*) –
- **menu_entries** (*Iterable[annize.features.distributables.flatpak.MenuEntry]*) –
- **environment** (*Dict[str, str]*) –
- **pkgrootpath** (*str*) –
- **pkgpath_share** (*str*) –
- **pkgpath_share_applications** (*str*) –
- **pkgpath_share_icons** (*str*) –
- **result** (annize.fs.FsEntry) –

Return type None

command: Optional[str]

environment: Dict[str, str]

filesystems: Iterable[str]

kitversion: Optional[str]

menu_entries: Iterable[*annize.features.distributables.flatpak.MenuEntry*]

name: str

pkgpath_share: str = None

pkgpath_share_applications: str = None

pkgpath_share_icons: str = None

pkgrootpath: str = None

platform: str

result: *annize.fs.FsEntry* = None

sdk: str

shares: Iterable[str]

sockets: Iterable[str]

source: *annize.fs.FsEntry*

_abc_impl = <_abc_data object>

_initialize()

classmethod _mkpackage(*build*)

Parameters build (annize.features.distributables.flatpak.FlatpakImage._BuildInfo) –

Return type *annize.fs.FsEntry*

classmethod _mkpackage_applysource(*build*)

Parameters build (annize.features.distributables.flatpak.FlatpakImage._BuildInfo) –

Return type None

classmethod _mkpackage_flatpak_build_export(*build*)

Parameters build (annize.features.distributables.flatpak.FlatpakImage._BuildInfo) –

Return type None

classmethod _mkpackage_flatpak_build_finish(*build*)

Parameters build (annize.features.distributables.flatpak.FlatpakImage._BuildInfo) –

Return type None

classmethod _mkpackage_flatpak_build_init(*build*)

Parameters build (annize.features.distributables.flatpak.FlatpakImage._BuildInfo) –

Return type None

classmethod _mkpackage_prepareinfos(*build*, *tmpdir*)

Parameters

- **build** (annize.features.distributables.flatpak.FlatpakImage._BuildInfo) –
- **tmpdir** (*str*) –

Return type None

classmethod _mkpackage_share(*build*)

Parameters build (annize.features.distributables.flatpak.FlatpakImage._BuildInfo) –

Return type None

class annize.features.distributables.flatpak.**FlatpakrefFile**(**, refname=None, package_name, title=None, branch='master', runtime_repository_url=None, gpgkey=None, repository*)

Bases: *annize.fs.AdHocFsEntry*

Parameters

- **refname** (*str*) –
- **package_name** (*str*) –
- **title** (*str*) –
- **branch** (*str*) –
- **runtime_repository_url** (*str*) –
- **gpgkey** (*bytes*) –
- **repository** (annize.features.distributables.flatpak.Repository) –

_abc_impl = <_abc_data object>

_initialize()

class annize.features.distributables.flatpak.**GpgFile**(**, refname=None, gpgkey=None*)

Bases: *annize.fs.AdHocFsEntry*

Parameters

- **refname** (*str*) –
- **gpgkey** (*bytes*) –

_abc_impl = <_abc_data object>

_initialize()

class annize.features.distributables.flatpak.**Group**(**, source, title, repository, package_name, project_short_hint_name=None, package_short_name=None*)

Bases: *annize.features.distributables.common.Group*

Parameters

- **source** (annize.fs.FsEntry) –
- **title** (*str*) –
- **repository** (annize.features.distributables.flatpak.Repository) –
- **package_name** (*str*) –
- **project_short_hint_name** (*str*) –
- **package_short_name** (*str*) –

property description

get_files()

class annize.features.distributables.flatpak.**LocalRepository**(**, public_url, friendly_name_suggestion=None, upload_path=None, upload_fsentry=None*)

Bases: *annize.features.distributables.flatpak.Repository*

Parameters

- **public_url** (*str*) –
- **friendly_name_suggestion** (*str*) –
- **upload_path** (*str*) –
- **upload_fsentry** (annize.fs.FsEntry) –

upload(*source*)

Parameters **source** (annize.fs.FsEntry) –

class annize.features.distributables.flatpak.**MenuEntry**(**, name, title, category, command, gui, icon*)

Bases: object

Parameters

- **name** (*str*) –
- **title** (*str*) –
- **category** (*Tuple*) –
- **command** (*str*) –
- **gui** (*bool*) –
- **icon** (annize.fs.FsEntry) –

class annize.features.distributables.flatpak.**Repository**(**, public_url, friendly_name_suggestion=None*)

Bases: object

Parameters

- **public_url** (*str*) –
- **friendly_name_suggestion** (*str*) –

property friendly_name_suggestion

property public_url

upload(*source*)

Parameters **source** (annize.fs.FsEntry) –

class annize.features.distributables.flatpak.**Share**(***b*)

Bases: object

annize.features.distributables.pythonwheel module

class annize.features.distributables.pythonwheel.**ExecutableLink**(*, *linkname*, *modulename*, *methodname*, *is_gui*)

Bases: object

Parameters

- **linkname** (*str*) –
- **modulename** (*str*) –
- **methodname** (*str*) –
- **is_gui** (*bool*) –

property is_gui

property linkname

property methodname

property modulename

class annize.features.distributables.pythonwheel.**Package**(*, *source*, *executable_links*, *packagename=None*, *description=None*, *homepage_url=None*, *long_description=None*, *version=None*, *keywords=None*, *license=None*, *author=None*)

Bases: *annize.fs.AdHocFsEntry*

Parameters

- **source** (annize.fs.FsEntry) –
- **executable_links** (*List[*annize.features.distributables.pythonwheel.ExecutableLink*]*) –
- **packagename** (*Optional[str]*) –
- **description** (*Optional[str]*) –
- **homepage_url** (*Optional[str]*) –
- **long_description** (*Optional[str]*) –

- **version** (annize.data.version.Version) –
- **keywords** (annize.features.base.Keywords) –
- **license** (annize.features.licensing.License) –
- **author** (annize.features.authors.Author) –

class _BuildInfo(*source: annize.fs.FsEntry*, *description: str*, *long_description: str*, *keywords: annize.features.base.Keywords*, *name: str*, *version: annize.data.version.Version*, *homepage: str*, *author: annize.features.authors.Author*, *license: annize.features.licensing.License*, *executable_links: List[annize.features.distributables.pythonwheel.ExecutableLink]*, *pkgrootpath: str = None*, *pkgpath_setuppy: str = None*, *setuppy_conf: Dict[str, Union[Any, NoneType]] = None*, *result: annize.fs.FsEntry = None*)

Bases: `object`

_BuildInfo(source: annize.fs.FsEntry, description: str, long_description: str, keywords: annize.features.base.Keywords, name: str, version: annize.data.version.Version, homepage: str, author: annize.features.authors.Author, license: annize.features.licensing.License, executable_links: List[annize.features.distributables.pythonwheel.ExecutableLink], pkgrootpath: str = None, pkgpath_setuppy: str = None, setuppy_conf: Dict[str, Union[Any, NoneType]] = None, result: annize.fs.FsEntry = None)

Parameters

- **source** (annize.fs.FsEntry) –
- **description** (*str*) –
- **long_description** (*str*) –
- **keywords** (annize.features.base.Keywords) –
- **name** (*str*) –
- **version** (annize.data.version.Version) –
- **homepage** (*str*) –
- **author** (annize.features.authors.Author) –
- **license** (annize.features.licensing.License) –
- **executable_links** (*List[*annize.features.distributables.pythonwheel.ExecutableLink*]*) –
- **pkgrootpath** (*str*) –
- **pkgpath_setuppy** (*str*) –
- **setuppy_conf** (*Dict[str, Optional[Any]]*) –
- **result** (annize.fs.FsEntry) –

Return type None

author: *annize.features.authors.Author*

description: **str**

executable_links: **List[***annize.features.distributables.pythonwheel.ExecutableLink***]**

homepage: **str**

keywords: *annize.features.base.Keywords*

license: *annize.features.licensing.License*

long_description: str

name: str

pkgpath_setuppy: str = None

pkgrootpath: str = None

result: ***annize.fs.FsEntry*** **= None**

setuppy_conf: Dict[str, Optional[Any]] = None

source: ***annize.fs.FsEntry***

version: ***annize.data.version.Version***

_abc_impl = <_abc_data object>

_initialize()

classmethod _mkpackage(*build*)

Parameters build (annize.features.distributables.pythonwheel.Package._BuildInfo) –

Return type *annize.fs.FsEntry*

classmethod _mkpackage_bdist_wheel(*build*)

Parameters build (annize.features.distributables.pythonwheel.Package._BuildInfo) –

Return type None

classmethod _mkpackage_mkexeclinks(*build*)

Parameters build (annize.features.distributables.pythonwheel.Package._BuildInfo) –

Return type None

classmethod _mkpackage_mkmanifestin(*build*)

Parameters build (annize.features.distributables.pythonwheel.Package._BuildInfo) –

Return type None

classmethod _mkpackage_mksetuppyconf(*build*)

Parameters build (annize.features.distributables.pythonwheel.Package._BuildInfo) –

Return type None

classmethod _mkpackage_prepareinfos(*build*, *tmpdir*)

Parameters

- **build** (annize.features.distributables.pythonwheel.Package._BuildInfo) –
- **tmpdir** (*str*) –

Return type None

classmethod _mkpackage_setuppyconf_classifiers(*build*)

Parameters **build** (annize.features.distributables.pythonwheel.Package._BuildInfo) –

Return type None

classmethod _mkpackage_setuppyconf_install_requires(*build*)

Parameters **build** (annize.features.distributables.pythonwheel.Package._BuildInfo) –

Return type None

classmethod _mkpackage_setuppyconf_prepare(*build*)

Parameters **build** (annize.features.distributables.pythonwheel.Package._BuildInfo) –

Return type None

annize.features.distributables.tar module

class annize.features.distributables.tar.**Package**(**, packagename=None, packagenamepostfix=None, source, version=None, license=None, documentation=None*)

Bases: *annize.fs.AdHocFsEntry*

Parameters

- **packagename** (*Optional[str]*) –
- **packagenamepostfix** (*Optional[str]*) –
- **source** (annize.fs.FsEntry) –
- **version** (annize.data.version.Version) –
- **license** (annize.features.licensing.License) –
- **documentation** (annize.fs.FsEntry) –

_abc_impl = <_abc_data object>

_initialize()

annize.features.documentation namespace

Subpackages

annize.features.documentation.sphinx namespace

Subpackages

annize.features.documentation.sphinx.output namespace

Submodules

annize.features.documentation.sphinx.output.common module

class annize.features.documentation.sphinx.output.common.**OutputGenerator**(*outputspec*)

Bases: abc.ABC

Base class for documentation output specifications. See render().

_abc_impl = <_abc_data object>

abstract formatname()

Returns the Sphinx format name.

Return type str

abstract classmethod is_compatible_for(*outputspec*)

Parameters outputspec (annize.features.documentation.common.OutputSpec) –

Return type bool

multilanguage_frame(*document*)

Parameters document (*documentationsphinx.Document*) –

Return type *annize.features.documentation.common._DocumentGenerateAllLanguagesResult*

property outputspec

postproc(*preresult*)

Parameters preresult (annize.fs.FsEntry) –

Return type *annize.fs.FsEntry*

prepare_generate(*geninfo*)

Parameters geninfo (*documentationsphinx.Document.GenerateInfo*) –

Return type None

annize.features.documentation.sphinx.output.common.**find_output_generator_for_outputspec**(*out*

annize.features.documentation.sphinx.output.common.**register_output_generator**(*outputgenerator*)

annize.features.documentation.sphinx.output.html module

class annize.features.documentation.sphinx.output.html.**HtmlOutputGenerator**(*outputspec*)

Bases: *annize.features.documentation.sphinx.output.common.OutputGenerator*

HTML documentation output.

_abc_impl = <_abc_data object>

formatname()

Returns the Sphinx format name.

classmethod is_compatible_for(*outputspec*)

multilanguage_frame(*document*)

prepare_generate(*geninfo*)

class annize.features.documentation.sphinx.output.html.**HtmlOutputSpec**(*, *is_homepage=False*, *title=None*, *short_title=None*, *short_desc=None*, *theme=None*, *masterlink=None*, *logo_image=None*, *background_image=None*)

Bases: *annize.features.documentation.common.HtmlOutputSpec*

Parameters

- **path** – The output path.
- **theme** (*Optional[str]*) – The sphinx html theme name.
- **homepagemode** – If to render output for a homepage with slight different stylings and behaviors.
- **title** (*Optional[str]*) – The html title.
- **short_title** (*Optional[str]*) – The short html title.
- **short_desc** (*Optional[str]*) – The short description. Ignored by most themes.
- **masterlink** (*Optional[str]*) – Url that overrides the target of the main heading (which is also a link).
- **is_homepage** (*bool*) –
- **logo_image** (*Optional[annize.fs.FsEntry]*) –
- **background_image** (*Optional[annize.fs.FsEntry]*) –

_abc_impl = <_abc_data object>

property background_image

property logo_image

property masterlink

property short_desc

property short_title

property theme

annize.features.documentation.sphinx.output.pdf module

class annize.features.documentation.sphinx.output.pdf.**PdfOutputGenerator**(*outputspec*)

Bases: *annize.features.documentation.sphinx.output.common.OutputGenerator*

PDF documentation output.

_abc_impl = <_abc_data object>

formatname()

Returns the Sphinx format name.

classmethod is_compatible_for(*outputspec*)

postproc(*preresult*)

resultpath(*projectname*, *buildpath*)

annize.features.documentation.sphinx.output.plaintext module

class annize.features.documentation.sphinx.output.plaintext.**PlaintextOutputGenerator**(*outputsp*

Bases: *annize.features.documentation.sphinx.output.common.OutputGenerator*

Plaintext documentation output.

_abc_impl = <_abc_data object>

formatname()

Returns the Sphinx format name.

classmethod is_compatible_for(*outputspec*)

Submodules

annize.features.documentation.sphinx._utils module

annize.features.documentation.sphinx._utils.**serialize_for_confpy**(*obj*)

Parameters **obj** (*Optional[Any]*) –

Return type str

annize.features.documentation.sphinx.common module

class annize.features.documentation.sphinx.common.**AboutProjectDocument**(**, projectname=None, authors=None, version=None, release=None, dependencies, project_version=None*)

Bases: *annize.features.documentation.sphinx.common.Document*

Parameters

- **projectname** (*Optional[str]*) –
- **authors** (*Optional[str]*) –
- **version** (*Optional[str]*) –
- **release** (*Optional[str]*) –
- **dependencies** (*List[*annize.features.dependencies.common.Dependency*]*) –
- **project_version** (annize.data.version.Version) –

_abc_impl = <_abc_data object>

_generate_master_doc(*geninfo*)

available_languages()

Return type List[str]

class annize.features.documentation.sphinx.common.**ApiReferenceDocument**(**, language, heading=None, source=None*)

Bases: *annize.features.documentation.sphinx.common.Document*

An api reference.

Parameters

- **language** (annize.features.documentation.sphinx.common.ApiReferenceLanguage) –
- **heading** (*Optional[str]*) –
- **source** (annize.fs.FsEntry) –

__get_refgeninfo(*geninfo*)

Parameters **geninfo** (annize.features.documentation.sphinx.common.Document.GenerateInfo) –

_abc_impl = <_abc_data object>

_generate_master_doc(*geninfo*)

_prepare_generate(*geninfo*)

available_languages()

Return type List[str]

class annize.features.documentation.sphinx.common.**ApiReferenceLanguage**

Bases: abc.ABC

Base class for a programming language in api references. See ApiReferencePiece.

class ApiReferenceGenerateInfo(*geninfo*, *source*, *heading*)

Bases: *annize.features.documentation.sphinx.common.Document.GenerateInfo*

Parameters

- **geninfo** (annize.features.documentation.sphinx.common.Document.GenerateInfo) –
- **source** (annize.fs.FsEntry) –
- **heading** (*str*) –

property heading

property source

_abc_impl = <_abc_data object>

abstract generate_master_doc(*rgeninfo*)

Parameters **rgeninfo** (annize.features.documentation.sphinx.common.ApiReferenceLanguage.ApiReferenceGenerateInfo) –

Return type str

abstract prepare_generate(*rgeninfo*)

Parameters **rgeninfo** (annize.features.documentation.sphinx.common.ApiReferenceLanguage.ApiReferenceGenerateInfo) –

Return type None

class annize.features.documentation.sphinx.common.**ArgparseCommandLineInterfaceDocument**(*, *parse factory*, *program name head ing=* *sourc*

Bases: *annize.features.documentation.sphinx.common.Document*

Parameters

- **parserfactory** (*str*) –
- **programname** (*str*) –

- **heading** (*Optional[str]*) –
- **source** (annize.fs.FsEntry) –

_abc_impl = <_abc_data object>

_generate_master_doc (*geninfo*)

_prepare_generate (*geninfo*)

available_languages ()

Return type List[str]

class annize.features.documentation.sphinx.common.**CompositeDocument** (**, projectname=None, authors=None, version=None, release=None, documents*)

Bases: *annize.features.documentation.sphinx.common.Document*

Parameters

- **projectname** (*Optional[str]*) –
- **authors** (*Optional[str]*) –
- **version** (*Optional[str]*) –
- **release** (*Optional[str]*) –
- **documents** (*Iterable[*annize.features.documentation.sphinx.common.Document*]*) –

__get_inner_generateinfo (*geninfo, innerdoc*)

Parameters

- **geninfo** (annize.features.documentation.sphinx.common.Document.GenerateInfo) –
- **innerdoc** (annize.features.documentation.sphinx.common.Document) –

_abc_impl = <_abc_data object>

_generate_master_doc (*geninfo*)

_prepare_generate (*geninfo*)

class annize.features.documentation.sphinx.common.**Document** (**, projectname=None, authors=None, version=None, release=None*)

Bases: *annize.features.documentation.common.Document*

Parameters

- **projectname** (*Optional[str]*) –

- **authors** (*Optional[str]*) –
- **version** (*Optional[str]*) –
- **release** (*Optional[str]*) –

class GenerateInfo(*intstruct*, *outdir*, *confdir*, *language*, ***, *configvalues=None*, *configlines=None*, *entry_path=''*)

Bases: `object`

Parameters

- **intstruct** (annize.fs.FsEntry) –
- **outdir** (annize.fs.FsEntry) –
- **confdir** (annize.fs.FsEntry) –
- **language** (*str*) –
- **configvalues** (*Optional[Dict]*) –
- **configlines** (*Optional[List]*) –
- **entry_path** (*str*) –

_to_dict()

property confdir

property configlines

property configvalues

property entry_path

property intstruct

property language

property outdir

_abc_impl = <_abc_data object>

property _authors

abstract _generate_master_doc(*geninfo*)

Parameters geninfo (annize.features.documentation.sphinx.common.Document.GenerateInfo) –

Return type str

_prepare_generate(*geninfo*)

Parameters geninfo (annize.features.documentation.sphinx.common.Document.GenerateInfo) –

Return type None

property _projectname

property _release

property _version

generate(*outputspec*, ***, *language='?'*)

generate_all_languages(*outputspec*)

class annize.features.documentation.sphinx.common.**ReadmeDocument**(**, projectname=None, authors=None, version=None, release=None, documents, title=None, dependencies, project_version=None*)

Bases: *annize.features.documentation.sphinx.common.CompositeDocument*

A reStructuredText formatted file or a directory of such files.

Parameters

- **projectname** (*Optional[str]*) –
- **authors** (*Optional[str]*) –
- **version** (*Optional[str]*) –
- **release** (*Optional[str]*) –
- **documents** (*Iterable[*annize.features.documentation.sphinx.common.Document*]*) –
- **title** (*Optional[str]*) –
- **dependencies** (*List[*annize.features.dependencies.common.Dependency*]*) –
- **project_version** (annize.data.version.Version) –

_abc_impl = <_abc_data object>

available_languages()

Return type List[str]

property title

class annize.features.documentation.sphinx.common.**RstDocument**(**, variants, **kwargs*)

Bases: *annize.features.documentation.sphinx.common.Document*

A reStructuredText formatted file or a directory of such files.

Parameters **variants** (*List[*annize.features.documentation.sphinx.common.RstDocumentVariant*]*) –

__get_variant(*language*)

Parameters **language** (*str*) –

Return type Optional[*annize.features.documentation.sphinx.common.RstDocumentVariant*]

_abc_impl = <_abc_data object>

_generate_master_doc(*geninfo*)

available_languages()

class annize.features.documentation.sphinx.common.**RstDocumentVariant**(**, language='?', source*)

Bases: object

Parameters

- **language** (*str*) –
- **source** (annize.fs.FsEntry) –

property language

property source

annize.features.documentation.sphinx.cpp module

class annize.features.documentation.sphinx.cpp.**CppApiReferenceLanguage**(***kwargs*)

Bases: *annize.features.documentation.sphinx.doxygencompat.DoxygenSupportedApiReferenceLanguage*

C++ language support for api references.

_abc_impl = <_abc_data object>

annize.features.documentation.sphinx.doxygencompat module

class annize.features.documentation.sphinx.doxygencompat.**DoxygenSupportedApiReferenceLangua**

Bases: *annize.features.documentation.sphinx.common.ApiReferenceLanguage*

Language support for api references powered by Doxygen.

Parameters

- **_doxygenopts** (*Optional[Dict[str, str]]*) –
- **extract_all** (*bool*) –
- **extract_private** (*bool*) –
- **extract_package** (*bool*) –
- **extract_static** (*bool*) –
- **extract_local_classes** (*bool*) –
- **extract_local_methods** (*bool*) –
- **extract_anon_nspaces** (*bool*) –
- **extract_priv_virtual** (*bool*) –
- **file_patterns** (*str*) –
- **inline_inherited_memb** (*bool*) –
- **inherit_docs** (*bool*) –

- **hide_undoc_members** (*bool*) –
- **hide_undoc_classes** (*bool*) –
- **exclude_patterns** (*str*) –
- **predefined** (*Optional[List[str]]*) –

__info(*outpath*)

_abc_impl = <_abc_data object>

_generate_master_doc(*srcpath*, *intstructpath*, *confdirpath*, *heading*)

prepare_generate(*rootpath*, *outpath*)

annize.features.documentation.sphinx.javascript module

class annize.features.documentation.sphinx.javascript.**JavaScriptApiReferenceLanguage**

Bases: *annize.features.documentation.sphinx.common.ApiReferenceLanguage*

JavaScript language support for api references.

__jsfiles(*dirf*)

Parameters **dirf** (*str*) –

Return type List[str]

__scanjsfile(*f*)

Parameters **f** (*str*) –

Return type str

_abc_impl = <_abc_data object>

generate_master_doc(*rgeninfo*)

prepare_generate(*rgeninfo*)

annize.features.documentation.sphinx.python module

class annize.features.documentation.sphinx.python.**Python3ApiReferenceLanguage**(**, show_undoc_mem show_protected_m exclude=None*)

Bases: *annize.features.documentation.sphinx.common.ApiReferenceLanguage*

Python 3 language support for api references.

Parameters

- **show_undoc_members** (*bool*) –
- **show_protected_members** (*bool*) –
- **exclude** (*Optional[List[str]]*) –

__patch_property_types_in_docstrings(*pdir*)

Parameters **pdir** (*str*) –

Return type None

_abc_impl = <_abc_data object>

generate_master_doc (*rgeninfo*)

prepare_generate (*rgeninfo*)

annize.features.documentation.sphinx.rst module

class annize.features.documentation.sphinx.rst.**RstGenerator**

Bases: `object`

Generator for some documentation source parts.

static heading (*text*, ***, *variant='='*, *sub=False*, *anchor=None*)

Generates documentation source for a section heading.

Parameters

- **text** (*str*) –
- **variant** (*str*) –
- **sub** (*bool*) –
- **anchor** (*Optional[str]*) –

Return type str

Submodules

annize.features.documentation.common module

class annize.features.documentation.common.**Document**

Bases: `abc.ABC`

_abc_impl = <_abc_data object>

abstract available_languages ()

Return type List[str]

abstract generate (*outputspec*, ***, *language='?'*)

Parameters language (*str*) –

Return type *annize.features.documentation.common._DocumentGenerateResult*

abstract generate_all_languages (*outputspec*)

Return type *annize.features.documentation.common._DocumentGenerateAllLanguagesResult*

class annize.features.documentation.common.**GeneratedDocument** (***, *document*, *outputspec*, *language=None*, *filename=None*)

Bases: *annize.fs.AdHocFsEntry*

Parameters

- **document** (annize.features.documentation.common.Document) –
- **outputspec** (annize.features.documentation.common.OutputSpec) –

- **language** (*str*) –
- **filename** (*str*) –

_abc_impl = <_abc_data object>

_initialize()

class annize.features.documentation.common.**HtmlOutputSpec**(*, *is_homepage=False*, *title=None*)

Bases: *annize.features.documentation.common.OutputSpec*

HTML documentation output.

Parameters

- **is_homepage** (*bool*) – If to render output for a homepage with slight different stylings and behavior.
- **title** (*Optional[str]*) – The html title. TODO higher level thing here?!

_abc_impl = <_abc_data object>

property is_homepage

property title

class annize.features.documentation.common.**OutputSpec**

Bases: abc.ABC

Base class for documentation output specifications. See render().

_abc_impl = <_abc_data object>

class annize.features.documentation.common.**PdfOutputSpec**

Bases: *annize.features.documentation.common.OutputSpec*

PDF documentation output.

_abc_impl = <_abc_data object>

class annize.features.documentation.common.**PlaintextOutputSpec**

Bases: *annize.features.documentation.common.OutputSpec*

Plaintext documentation output.

_abc_impl = <_abc_data object>

class annize.features.documentation.common.**_DocumentGenerateAllLanguagesResult**(*file*, *entry_path*, *entry_paths_for_la*

Bases: *annize.features.documentation.common._DocumentGenerateResult*

Parameters

- **file** (annize.fs.FsEntry) –
- **entry_path** (*str*) –
- **entry_paths_for_languages** (*Dict[str, str]*) –

entry_path_for_language(*language*)

Parameters language (*str*) –

Return type str

property languages

class annize.features.documentation.common.**_DocumentGenerateResult**(*file*, *entry_path*)

Bases: object

Parameters

- **file** (annize.fs.FsEntry) –
- **entry_path** (*str*) –

_set_entry_path(*entry_path*)

Parameters **entry_path** (*str*) –

Return type None

property entry_path

property file

annize.features.files namespace

Subpackages

annize.features.files.transfer namespace

Submodules

annize.features.files.transfer.common module

class annize.features.files.transfer.common.**Endpoint**

Bases: object

access_filesystem(*rootpath*)

Parameters **rootpath** (*str*) –

Return type AbstractContextManager[*annize.fs.FsEntry*]

class annize.features.files.transfer.common.**FsEndpoint**(*, *fsentry=None*, *path=None*)

Bases: *annize.features.files.transfer.common.Endpoint*

Parameters

- **fsentry** (annize.fs.FsEntry) –
- **path** (*str*) –

access_filesystem(*rootpath*)

Parameters **rootpath** (*str*) –

Return type AbstractContextManager[*annize.fs.FsEntry*]

class annize.features.files.transfer.common.**Upload**(*, *source*, *destination_endpoint*, *destination_path*)

Bases: object

Parameters

- **source** (annize.fs.FsEntry) –
- **destination_endpoint** (annize.features.files.transfer.common.Endpoint) –
- **destination_path** (*str*) –

annize.features.files.transfer.ssh module

class annize.features.files.transfer.ssh.**Endpoint**(**, host, port=22, username, identity_file=None, has_shell_access=False*)

Bases: *annize.features.files.transfer.common.Endpoint*

Parameters

- **host** (*str*) –
- **port** (*int*) –
- **username** (*str*) –
- **identity_file** (*Optional[str]*) –
- **has_shell_access** (*bool*) –

access_filesystem(*rootpath*)

exec(*cmdline*)

Parameters cmdline (*str*) –

Return type *TODO*

property has_shell_access

property host

property identity_file

locationstring(*path=None*)

Parameters path (*Optional[str]*) –

Return type str

property port

property username

Submodules

annize.features.files.common module

class annize.features.files.common.**Dir**(**, excludes, parts, root=None, path=None, name=None*)

Bases: *annize.features.files.common.FsEntry*

Parameters

- **excludes** (*Iterable[*annize.features.files.common.Exclude*]*) –
- **parts** (*Iterable[*annize.features.files.common.DirPart*]*) –

- **root** (annize.fs.FsEntry) –
- **path** (*str*) –
- **name** (*str*) –

classmethod __transfer_predicate_and(**predicates*)

Parameters **predicates** (*Callable[[str,* FsEntry*,* FsEntry*], bool]*) –

Return type Callable[[str, *annize.features.files.common.FsEntry*, *annize.features.files.common.FsEntry*], bool]

classmethod __transfer_predicate_for_exclude(*exclude*)

Parameters **exclude** (annize.features.files.common.Exclude) –

Return type Callable[[str, *annize.features.files.common.FsEntry*, *annize.features.files.common.FsEntry*], bool]

classmethod __transfer_predicate_for_excludes(*excludes*)

Parameters **excludes** (*Iterable[*annize.features.files.common.Exclude*]*) –

Return type Callable[[str, *annize.features.files.common.FsEntry*, *annize.features.files.common.FsEntry*], bool]

_abc_impl = <_abc_data object>

_initialize()

class annize.features.files.common.**DirPart**(**, excludes=None, root=None, source_path=None, destination_path=None, path=None, destination_is_parent=False*)

Bases: object

Parameters

- **excludes** (*Iterable[*annize.features.files.common.Exclude*]*) –
- **root** (annize.fs.FsEntry) –
- **source_path** (*str*) –
- **destination_path** (*str*) –
- **path** (*str*) –
- **destination_is_parent** (*bool*) –

property destination_is_parent

property destination_path

property excludes

property path

property root

class annize.features.files.common.**Exclude**(**, by_path_pattern=None, by_path=None, by_name_pattern=None, by_name=None*)

Bases: object

Parameters

- **by_path_pattern** (*str*) –

- **by_path** (*str*) –
- **by_name_pattern** (*str*) –
- **by_name** (*str*) –

__does_exclude (*by_text*, *by_pattern*)

Parameters

- **text** (*str*) –
- **by_text** (*str*) –
- **by_pattern** (*re.Pattern*) –

Return type bool

does_exclude (*relative_path*, *source*, *destination*)

Parameters

- **relative_path** (*str*) –
- **source** (annize.fs.FsEntry) –
- **destination** (annize.fs.FsEntry) –

Return type bool

class annize.features.files.common.**File** (***, *root=None*, *path=None*)

Bases: *annize.features.files.common.FsEntry*

Parameters

- **root** (annize.fs.FsEntry) –
- **path** (*str*) –

_abc_impl = <_abc_data object>

class annize.features.files.common.**FsEntry** (***, *root=None*, *path=None*)

Bases: *annize.fs.AdHocFsEntry*

Parameters

- **root** (annize.fs.FsEntry) –
- **path** (*str*) –

_abc_impl = <_abc_data object>

_initialize ()

property relative_path

property root

class annize.features.files.common.**MachineRootDirectory**

Bases: *annize.fs.AdHocFsEntry*

_abc_impl = <_abc_data object>

_initialize ()

class annize.features.files.common.**ProjectDir**

Bases: *annize.fs.AdHocFsEntry*

_abc_impl = <_abc_data object>

_initialize()

annize.features.homepage namespace

Subpackages

annize.features.homepage.sections namespace

Submodules

annize.features.homepage.sections.about module

class annize.features.homepage.sections.about.**Section**(**, head=<annize.i18n.ProvidedTrStr object>, sortindex=10000*)

Bases: *annize.features.homepage.common.HomepageSection*

_generate(*info*)

annize.features.homepage.sections.changelog module

class annize.features.homepage.sections.changelog.**Section**(**, changelog=None, head=<annize.i18n.ProvidedTrStr object>, sortindex=60000*)

Bases: *annize.features.homepage.common.HomepageSection*

Parameters **changelog** (annize.features.changelog.common.Changelog) –

_generate(*info*)

annize.features.homepage.sections.documentation module

class annize.features.homepage.sections.documentation.**Section**(**, documentation, head=<annize.i18n.ProvidedTrStr object>, sortindex=30000*)

Bases: *annize.features.homepage.common.HomepageSection*

Parameters **documentation** (*List*[annize.features.documentation.common.Document]) –

_generate(*info*)

preproc_generate(*info*)

annize.features.homepage.sections.download module

class annize.features.homepage.sections.download.**Section**(**, distributables, dependencies, head=<annize.i18n.ProvidedTrStr object>, sortindex=40000*)

Bases: *annize.features.homepage.common.HomepageSection*

Parameters

- **distributables** (*List[annize.features.distributables.common.Group]*) –
- **dependencies** (*List[annize.features.dependencies.common.Dependency]*) –

__generate_packagelist(*info*)

Parameters **info** (annize.features.homepage.common.HomepageSection._GenerateInfo) –

_generate(*info*)

postproc_generate(*info*)

preproc_generate(*info*)

annize.features.homepage.sections.download.**filehash**(*filepath*)

Parameters **filepath** (*str*) –

Return type str

annize.features.homepage.sections.download.**friendly_filesize**(*isize*)

Parameters **isize** (*int*) –

Return type str

annize.features.homepage.sections.gallery module

class annize.features.homepage.sections.gallery.**Section**(**, head=<annize.i18n.ProvidedTrStr object>, sortindex=50000, mediagalleries*)

Bases: *annize.features.homepage.common.HomepageSection*

Parameters **mediagalleries** (*List[annize.features.mediagalleries.Gallery]*) –

_generate(*info*)

preproc_generate(*info*)

annize.features.homepage.sections.imprint module

class annize.features.homepage.sections.imprint.**Section**(**, imprint, head=<annize.i18n.ProvidedTrStr object>, sortindex=70000*)

Bases: *annize.features.homepage.common.HomepageSection*

Parameters **imprint** (*str*) –

_generate(*info*)

annize.features.homepage.sections.license module

class annize.features.homepage.sections.license.**Section**(**, head=<annize.i18n.ProvidedTrStr object>, sortindex=20000*)

Bases: *annize.features.homepage.common.HomepageSection*

_generate(*info*)

Submodules

annize.features.homepage.common module

class annize.features.homepage.common.**GeneratedHomepage**(**, homepage*)

Bases: *annize.fs.AdHocFsEntry*

Parameters **homepage** (annize.features.homepage.common.Homepage) –

_abc_impl = <_abc_data object>

_initialize()

class annize.features.homepage.common.**Homepage**(**, title=None, short_desc=None, sections, languages='?'*)

Bases: object

Parameters

- **title** (*Optional[str]*) –
- **short_desc** (*Optional[str]*) –
- **sections** (*List[*annize.features.homepage.common.HomepageSection*]*) –
- **languages** (*str*) –

__generate_postpreproc(**, is_pre, document_root_directory, document_root_url*)

Parameters

- **customargs** (*Dict*) –
- **is_pre** (*bool*) –
- **document_root_directory** (annize.fs.FsEntry) –
- **document_root_url** (*str*) –

`__generate_section`(**, customargs, culture, document_root_directory, document_root_url, document_variant_directory*)

Parameters

- **`customargs`** (*`Dict`*) –
- **`culture`** (`annize.i18n.Culture`) –
- **`document_root_directory`** (`annize.fs.FsEntry`) –
- **`document_root_url`** (*`str`*) –
- **`document_variant_directory`** (`annize.fs.FsEntry`) –

`_append_section`(*section*)

Parameters **`section`** (`annize.features.homepage.common.HomepageSection`) –

Return type None

`generate`()

`property languages`

`property sections`

`property short_desc`

`property title`

`class` `annize.features.homepage.common.`**`HomepageSection`**(**, head, sortindex=0*)

Bases: `object`

Parameters

- **`head`** (*`str`*) –
- **`sortindex`** (*`int`*) –

`class Product`(**, rst_text='', media_files=()*)

Bases: `object`

Parameters

- **`rst_text`** (*`str`*) –
- **`media_files`** (*`List[`*`annize.fs.FsEntry`*`]`*) –

`append_rst`(*rst_text*)

Return type None

`attach_media_file`(*file*)

Parameters **`file`** (`annize.fs.FsEntry`) –

Return type None

`property media_files`

`property rst_text`

`class _GenerateInfo`(*culture: annize.i18n.Culture, customarg: Union[object, NoneType], document_root_url: str, document_variant_directory: annize.fs.FsEntry, document_variant_url: str*)

Bases: `object`

_GenerateInfo(culture: annize.i18n.Culture, customarg: Union[object, NoneType], document_root_url: str, document_variant_directory: annize.fs.FsEntry, document_variant_url: str)

Parameters

- **culture** (annize.i18n.Culture) –
- **customarg** (*Optional[object]*) –
- **document_root_url** (*str*) –
- **document_variant_directory** (annize.fs.FsEntry) –
- **document_variant_url** (*str*) –

Return type None

culture: *annize.i18n.Culture*

customarg: **Optional[object]**

document_root_url: **str**

document_variant_directory: *annize.fs.FsEntry*

document_variant_url: **str**

class _PrePostprocGenerateInfo(*document_root_directory:* annize.fs.FsEntry, *document_root_url: str*, *customarg: Union[object, NoneType]*)

Bases: object

_PrePostprocGenerateInfo(document_root_directory: annize.fs.FsEntry, document_root_url: str, customarg: Union[object, NoneType])

Parameters

- **document_root_directory** (annize.fs.FsEntry) –
- **document_root_url** (*str*) –
- **customarg** (*Optional[object]*) –

Return type None

customarg: **Optional[object]**

document_root_directory: *annize.fs.FsEntry*

document_root_url: **str**

_generate(*info*)

Parameters **info** (annize.features.homepage.common.HomepageSection._GenerateInfo) –

Return type Union[*annize.features.homepage.common.HomepageSection.Product*, str]

generate(*info*)

Parameters **info** (annize.features.homepage.common.HomepageSection._GenerateInfo) –

Return type Optional[*annize.features.homepage.common.HomepageSection.Product*]

property head

postproc_generate(*info*)

Parameters **info** (annize.features.homepage.common.HomepageSection._PrePostprocGenerateInfo) –

Return type None

preproc_generate(*info*)

Parameters **info** (annize.features.homepage.common.HomepageSection._PrePostprocGenerateInfo) –

Return type None

property sortindex

class annize.features.homepage.common.**PinoHomepageAuxInfo**

Bases: *annize.features.homepage.common.HomepageSection*

_generate(*info*)

preproc_generate(*info*)

class annize.features.homepage.common.**SimpleProjectHomepage**(**, changelog=None, dependencies, distributables, documentation, imprint=None, mediagalleries, **kwargs*)

Bases: *annize.features.homepage.common.Homepage*

Parameters

- **changelog** (annize.features.changelog.common.Changelog) –
- **dependencies** (*List[*annize.features.dependencies.common.Dependency*]*) –
- **distributables** (*List[*annize.features.distributables.common.Group*]*) –
- **documentation** (*List[*annize.features.documentation.common.Document*]*) –
- **imprint** (*str*) –
- **mediagalleries** (*List[*annize.features.mediagalleries.Gallery*]*) –

annize.features.i18n namespace

Submodules

annize.features.i18n.common module

class annize.features.i18n.common.**ProjectDefinedTranslationProvider**

Bases: *annize.i18n.TranslationProvider*

__translations_for_stringname(*stringname*)

Parameters **stringname** (*str*) –

Return type Dict[str, str]

add_translations(*stringname, variants*)

Parameters

- **stringname** (*str*) –

- **variants** (*Dict[str, str]*) –

Return type None

translate (*stringname*, *, *culture*)

class annize.features.i18n.common.**String** (*, *stringname=None*, *stringtr=None*, ***variants*)

Bases: *annize.i18n.ProvidedTrStr*

Parameters

- **stringname** (*Optional[str]*) –
- **stringtr** (*Optional[str]*) –
- **variants** (*str*) –

annize.features.i18n.gettext module

class annize.features.i18n.gettext.**GenerateMOs** (*, *po_directory*, *mo_directory*)

Bases: object

Parameters

- **po_directory** (annize.fs.FsEntry) –
- **mo_directory** (annize.fs.FsEntry) –

class annize.features.i18n.gettext.**UpdatePOs** (*, *po_directory*)

Bases: object

Parameters **po_directory** (annize.fs.FsEntry) –

annize.features.testing namespace

Submodules

annize.features.testing.common module

class annize.features.testing.common.**RunTests** (***b*)

Bases: object

class annize.features.testing.common.**Test**

Bases: object

run ()

class annize.features.testing.common.**TestGroup** (*, *tests*)

Bases: *annize.features.testing.common.Test*

Parameters **tests** (*List[*annize.features.testing.common.Test*]*) –

run ()

annize.features.testing.pylint module

class annize.features.testing.pylint.**Test**(*, *src*)

Bases: *annize.features.testing.common.Test*

Parameters **src** (annize.fs.FsEntry) –

run()

annize.features.testing.pytest module

class annize.features.testing.pytest.**Test**(*, *testdir*, *src*)

Bases: *annize.features.testing.common.Test*

Parameters

- **testdir** (*str*) –
- **src** (*str*) –

run()

annize.features.testing.pyunit module

class annize.features.testing.pyunit.**Test**(*, *src*)

Bases: *annize.features.testing.common.Test*

Parameters **src** (*str*) –

run()

annize.features.versioncontrol namespace

Submodules

annize.features.versioncontrol.common module

class annize.features.versioncontrol.common.**BuildVersion**(*, *base_version*, *vcs*)

Bases: *annize.data.version.Version*

Parameters

- **base_version** (annize.data.version.Version) –
- **vcs** (annize.features.versioncontrol.common.VersionControlSystem) –

__effversion()

property segments_tuples

property text

class annize.features.versioncontrol.common.**VersionControlSystem**

Bases: object

get_commit_message(*revision*)

Parameters **revision** (*str*) –

Return type str

get_commit_time(*revision*)

Parameters **revision** (*str*) –

Return type datetime.datetime

get_current_revision()

Return type str

get_revision_list()

Return type List[str]

get_revision_number(*revision*)

Parameters **revision** (*str*) –

Return type int

annize.features.versioncontrol.common.**default_version_control_system**()

Return type Optional[*annize.features.versioncontrol.common.VersionControlSystem*]

annize.features.versioncontrol.git module

class annize.features.versioncontrol.git.**ExcludeByGitIgnores**

Bases: *annize.features.files.common.Exclude*

does_exclude(*relative_path*, *source*, *destination*)

class annize.features.versioncontrol.git.**VersionControlSystem**(**, *path=''*)

Bases: *annize.features.versioncontrol.common.VersionControlSystem*

Parameters **path** (*str*) –

__call_git(*cmdline*)

Parameters **cmdline** (*List[str]*) –

Return type str

get_commit_message(*revision*)

get_commit_time(*revision*)

get_current_revision()

get_revision_list()

get_revision_number(*revision*)

property path

Submodules

annize.features.authors module

class annize.features.authors.**Author**(*, *fullname*, *email=None*)

Bases: `object`

Parameters

- **fullname** (*str*) –
- **email** (*str*) –

property email

property fullname

annize.features.authors.**default_project_authors**()

Return type List[*annize.features.authors.Author*]

annize.features.base module

class annize.features.base.**Basket**(*, *objects*)

Bases: *annize.data.container.Basket*

Parameters **objects** (*List[object]*) –

class annize.features.base.**BrandColor**(*, *red*, *green*, *blue*)

Bases: *annize.data.color.Color*

Parameters

- **red** (*float*) –
- **green** (*float*) –
- **blue** (*float*) –

class annize.features.base.**Data**(*, *project_name=None*, *pretty_project_name=None*, *summary=None*, *long_description=None*, *homepage_url=None*, *imprint=None*, *project_directory=None*)

Bases: `object`

Parameters

- **project_name** (*Union[str, annize.i18n.TrStr]*) –
- **pretty_project_name** (*Union[str, annize.i18n.TrStr]*) –
- **summary** (*Union[str, annize.i18n.TrStr]*) –
- **long_description** (*Union[str, annize.i18n.TrStr]*) –
- **homepage_url** (*Union[str, annize.i18n.TrStr]*) –
- **imprint** (*Union[str, annize.i18n.TrStr]*) –
- **project_directory** (*str*) –

property homepage_url

property imprint

property long_description

property pretty_project_name

property project_directory

property project_name

property summary

class annize.features.base.**DateTime**(**, iso=None*)

Bases: datetime.datetime

Parameters **iso** (*str*) –

class annize.features.base.**FirstOf**(**, objects*)

Bases: *annize.data.container.Basket*

Parameters **objects** (*List[object]*) –

class annize.features.base.**Keyword**(*text*)

Bases: *annize.features.base.Keywords*

Parameters **text** (*str*) –

class annize.features.base.**Keywords**(**, from_string='', split_by=' ', keywords=()*)

Bases: object

Parameters

- **from_string** (*str*) –
- **split_by** (*str*) –
- **keywords** (*List[str]*) –

property keywords

annize.features.base.**_get_data**(*key, defaultvalue*)

Parameters

- **key** (*str*) –
- **defaultvalue** (*Optional[Any]*) –

Return type Optional[Any]

annize.features.base.**brand_color**(**, none_on_undefined=False*)

Parameters **none_on_undefined** (*bool*) –

Return type *annize.data.color.Color*

annize.features.base.**homepage_url**()

Return type str

annize.features.base.**imprint**()

Return type str

annize.features.base.**long_description**()

Return type str

annize.features.base.**pretty_project_name**()

Return type str

annize.features.base.**project_directory**()

Return type str

`annize.features.base.`**`project_keywords`**`()`

Return type *annize.features.base.Keywords*

`annize.features.base.`**`project_name`**`()`

Return type str

`annize.features.base.`**`summary`**`()`

Return type str

annize.features.licensing module

`annize.features.licensing.`**`AFLv3`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`annize.features.licensing.`**`AGPLv3`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`annize.features.licensing.`**`Apache2`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`annize.features.licensing.`**`Artistic1`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`annize.features.licensing.`**`BSD2clause`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`annize.features.licensing.`**`BSD3clause`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`annize.features.licensing.`**`Cc0v1`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`annize.features.licensing.`**`CcBy3`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`annize.features.licensing.`**`CcByNc3`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`annize.features.licensing.`**`CcByNcNd3`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`annize.features.licensing.`**`CcByNcSa3`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`annize.features.licensing.`**`CcByNd3`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`annize.features.licensing.`**`CcBySa3`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`annize.features.licensing.`**`GPLv3`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`annize.features.licensing.`**`LGPLv3`**
alias of `annize.features.licensing._license.<locals>.ALicense`

`class` `annize.features.licensing.`**`License`**`(*, `*name*`, `*text=None*`, `***additional_info*`)`
Bases: `object`

Parameters

- **name** (*str*) –
- **text** (*str*) –

additional_info (*key*, *, *default=None*)

Parameters

- **key** (*str*) –
- **default** (*Optional[Any]*) –

Return type Any

property name

property text

annize.features.licensing.**MIT**
alias of annize.features.licensing._license.<locals>.ALicense

annize.features.licensing.**MPLv11**
alias of annize.features.licensing._license.<locals>.ALicense

annize.features.licensing.**MPLv2**
alias of annize.features.licensing._license.<locals>.ALicense

annize.features.licensing.**PublicDomain**
alias of annize.features.licensing._license.<locals>.ALicense

annize.features.licensing.**_license** (*name*)

Parameters **name** (*str*) –

Return type Type[*annize.features.licensing.License*]

annize.features.licensing.**project_licenses** ()

Return type List[*annize.features.licensing.License*]

annize.features.mediagalleries module

class annize.features.mediagalleries.**Gallery** (*, *source*, *title=''*)
Bases: object

Parameters

- **source** (annize.fs.FsEntry) –
- **title** (*str*) –

class Item (*file*, *description*)
Bases: object

property description

property file

_description_for_mediafile (*itemfile*)

Parameters **itemfile** (annize.fs.FsEntry) –

Return type str

property items

property title

annize.features.task module

class annize.features.task.**Task**(**, innertasks*)

Bases: object

Parameters **innertasks** (*List[object]*) –

annize.features.version module

class annize.features.version.**CommonVersionPattern**

Bases: *annize.data.version.VersionPattern*

class annize.features.version.**Line**(**, version*)

Bases: object

Parameters **version** (annize.data.version.Version) –

property version

class annize.features.version.**Version**(**, text=None, pattern=None, **segment_values*)

Bases: *annize.data.version.Version*

Parameters

- **text** (*str*) –
- **pattern** (annize.data.version.VersionPattern) –

annize.features.version.**default_version_pattern**()

Return type *annize.data.version.VersionPattern*

annize.features.version.**project_versions**()

Return type List[*annize.data.version.Version*]

annize.fs package

class annize.fs.**AdHocFsEntry**

Bases: *annize.fs.FsEntry*

__getinner()

_abc_impl = <_abc_data object>

_initialize()

Return type *annize.fs.FsEntry*

readable_path()

exception annize.fs.**BadDisowningError**(*path*)

Bases: RuntimeError

Parameters **path** (*str*) –

class annize.fs.**ChDir**(*path*)

Bases: object

Temporarily changes the current working directory in a scope of the `with` keyword.

Parameters **path** (*str*) – The path to the new working directory.

class annize.fs.**FsEntry**(**, is_ownable=False, is_owned=False*)

Bases: abc.ABC

Parameters

- **is_ownable** (*bool*) –
- **is_owned** (*bool*) –

_abc_impl = <_abc_data object>

basename()

Return type str

child_by_relative_path(*name*)

Return type *annize.fs.FsEntry*

children()

Return type List[*annize.fs.FsEntry*]

copy_to(*path, *, path_is_parent=False, merge=False, overwrite=False, transfer_predicate=None*)

Parameters

- **path** (*str*) –
- **path_is_parent** (*bool*) –
- **merge** (*bool*) –
- **overwrite** (*bool*) –
- **transfer_predicate** (*Optional[Callable[[str, FsEntry, FsEntry], bool]]*) –

Return type *annize.fs.FsEntry*

created_time()

Return type datetime.datetime

disowned()

Return type *annize.fs.FsEntry*

exists()

Return type bool

filesize()

Return type int

is_dir(*follow_symlinks=True*)

Parameters **follow_symlinks** (*bool*) –

Return type bool

is_file(*follow_symlinks=True*)

Parameters **follow_symlinks** (*bool*) –

Return type bool

is_link()

Return type bool

property is_ownable

property is_owned

mkdir(*, *exist_ok=True*)

Parameters **exist_ok** (*bool*) –

Return type None

modified_time()

Return type datetime.datetime

move_to(*path*, *, *path_is_parent=False*, *merge=False*, *overwrite=False*, *transfer_predicate=None*)

Parameters

- **path** (*str*) –
- **path_is_parent** (*bool*) –
- **merge** (*bool*) –
- **overwrite** (*bool*) –
- **transfer_predicate** (*Optional[Callable[[str, FsEntry, FsEntry], bool]]*) –

Return type *annize.fs.FsEntry*

owned(*allow_clone=True*)

Parameters **allow_clone** (*bool*) –

Return type *annize.fs.FsEntry*

parent()

Return type Optional[*annize.fs.FsEntry*]

path()

Return type str

read_file()

Return type bytes

readable_path()

Return type str

remove()

Return type None

temp_clone(*, *temprootpath=None*, *basename=None*)

Parameters

- **temprootpath** (*Optional[str]*) –
- **basename** (*Optional[str]*) –

Return type *annize.fs.FsEntry*

write_file(*content*)

Parameters **content** (*AnyStr*) –

Return type None

exception annize.fs.**MissingOwnershipError**(*path*)

Bases: RuntimeError

Parameters **path** (*str*) –

class annize.fs.**Mount**(*src*, *dst*, **, *options=()*, *mounter=None*, *unmounter=None*)

Bases: object

Parameters

- **src** (*str*) –
- **dst** (*str*) –
- **options** (*Iterable[str]*) –
- **mounter** (*Optional[Iterable[str]]*) –
- **unmounter** (*Optional[Iterable[str]]*) –

property destination

exception annize.fs.**OriginalOwnershipImpossibleError**(*path*)

Bases: RuntimeError

Parameters **path** (*str*) –

class annize.fs.**_SimpleFsEntry**(*path*, **, *is_ownable=False*, *is_owned=False*)

Bases: *annize.fs.FsEntry*

Parameters

- **path** (*str*) –
- **is_ownable** (*bool*) –
- **is_owned** (*bool*) –

_abc_impl = <_abc_data object>

readable_path()

class annize.fs.**_TransferHelper**

Bases: object

_TransferHelper__transfer_dirmerge(*action*, *transfer_predicate*)

Parameters

- **action** (*Callable*) –
- **transfer_predicate** (*Optional[Callable[[str, FsEntry, FsEntry], bool]]*) –

Return type Callable

_TransferHelper__transfer_prepare(*srcf*, *dstf*, *overwrite*)

Parameters

- **srcf** (annize.fs.FsEntry) –
- **dstf** (annize.fs.FsEntry) –
- **overwrite** (*bool*) –

Return type None

transfer_action_copy(*srcf*, *dstf*, *overwrite*)

Parameters

- **srcf** (annize.fs.FsEntry) –
- **dstf** (annize.fs.FsEntry) –
- **overwrite** (*bool*) –

Return type None

transfer_action_move(*srcf*, *dstf*, *overwrite*)

Parameters

- **srcf** (annize.fs.FsEntry) –
- **dstf** (annize.fs.FsEntry) –
- **overwrite** (*bool*) –

Return type None

transfer_to(*srcf*, *path*, ***, *merge*, *overwrite*, *action*, *transfer_predicate=None*)

Parameters

- **srcf** (annize.fs.FsEntry) –
- **path** (*str*) –
- **merge** (*bool*) –
- **overwrite** (*bool*) –
- **action** (*Callable*) –
- **transfer_predicate** (*Optional[Callable[[str, FsEntry, FsEntry], bool]]*) –

Return type *annize.fs.FsEntry*

annize.fs.**fsentry_by_path**(*path*, ***, *is_ownable=False*, *is_owned=False*)

Parameters

- **path** (*str*) –
- **is_ownable** (*bool*) –
- **is_owned** (*bool*) –

Return type *annize.fs.FsEntry*

Submodules

annize.fs.temp module

class annize.fs.temp.**TempFileByContent**(*, *content*, *filename='afile'*)
Bases: *annize.fs.AdHocFsEntry*

Parameters

- **content** (*AnyStr*) –
- **filename** (*Optional[str]*) –

_abc_impl = <_abc_data object>

_initialize()

Return type *annize.fs.FsEntry*

class annize.fs.temp.**_TempDir**(*dirname=None*, *temprootpath=None*)
Bases: *annize.fs.FsEntry*

Parameters

- **dirname** (*Optional[str]*) –
- **temprootpath** (*Optional[str]*) –

_TempDir__cleanup()

_abc_impl = <_abc_data object>

readable_path()

restrict_path(*path*)

annize.fs.temp.**fresh_temp_dir**(*dirname=None*, *, *temprootpath=None*)

Parameters

- **dirname** (*Optional[str]*) –
- **temprootpath** (*Optional[str]*) –

Return type *annize.fs.temp._TempDir*

annize.i18n package

class annize.i18n.**Culture**(*english_lang_name*, *iso_639_1_lang_code*)
Bases: object

Parameters

- **english_lang_name** (*str*) –
- **iso_639_1_lang_code** (*str*) –

property english_lang_name

static get_from_iso_639_1_lang_code(*iso_639_1_lang_code*)

property iso_639_1_lang_code

class annize.i18n.**GettextTranslationProvider**(*mopath*)
Bases: *annize.i18n.TranslationProvider*

translate (*stringname*, *, *culture*)

Parameters

- **stringname** (*str*) –
- **culture** (annize.i18n.Culture) –

class annize.i18n.**IndependentTrStr** (***variants*)

Bases: *annize.i18n.TrStr*

Parameters **variants** (*str*) –

translate (*culture=None*)

Return type str

exception annize.i18n.**NoCurrentCultureError**

Bases: TypeError

class annize.i18n.**ProvidedTrStr** (*stringname*)

Bases: *annize.i18n.TrStr*

Parameters **stringname** (*str*) –

translate (*culture=None*)

Return type str

class annize.i18n.**TODO** (*txt*)

Bases: *annize.i18n.TrStr*

translate (*culture=None*)

Parameters **culture** (*Optional[Union[*Culture*, str]]*) –

Return type str

class annize.i18n.**TrStr**

Bases: object

format (**args*, ***kwargs*)

Return type *annize.i18n.TrStr*

static tr (*stringname*)

Parameters **stringname** (*str*) –

Return type *annize.i18n.TrStr*

translate (*culture=None*)

Parameters **culture** (*Optional[Union[*Culture*, str]]*) –

Return type str

annize.i18n.**TrStrOrStr**

Type annotation for something that can be either a str or a *TrStr*.

alias of Union[str, annize.i18n.TrStr]

class annize.i18n.**TranslationProvider**

Bases: object

translate (*stringname*, *, *culture*)

Parameters

- **stringname** (*str*) –
- **culture** (annize.i18n.Culture) –

Return type str

exception annize.i18n.**TranslationUnavailableError**(*stringname*, *language*)

Bases: TypeError

Parameters

- **stringname** (*str*) –
- **language** (*str*) –

annize.i18n.**add_translation_provider**(*provider*, ***, *priority=0*)

Parameters

- **provider** (annize.i18n.TranslationProvider) –
- **priority** (*int*) –

Return type None

annize.i18n.**annize_user_interaction_culture**()

Return type *annize.i18n.Culture*

annize.i18n.**current_culture**()

Return type *annize.i18n.Culture*

annize.i18n.**get_culture**(*culture*)

Parameters **culture** (*Optional[Union[Culture, str]]*) –

Return type Optional[*annize.i18n.Culture*]

annize.i18n.**tr**(*stringname*, ***, *culture=None*)

Parameters

- **stringname** (*str*) –
- **culture** (*Optional[Union[Culture, str]]*) –

Return type str

annize.i18n.**tr_if_trstr**(*txt*, *culture=None*)

Parameters

- **txt** (*Union[str, annize.i18n.TrStr]*) –
- **culture** (*Optional[Union[Culture, str]]*) –

Return type str

annize.i18n.**translation_providers**()

Return type List[*annize.i18n.TranslationProvider*]

annize.project package

Definitions of Annize projects.

class annize.project.**ArgumentNode**

Bases: *annize.project.Node*

_str_helper()

property append_to

property arg_name

property name

exception annize.project.**BadStructureError**(*message*)

Bases: ValueError

Parameters **message** (*str*) –

class annize.project.**BlockNode**

Bases: *annize.project.Node*

classmethod _allowed_child_types()

class annize.project.**CallableNode**

Bases: *annize.project.ArgumentNode*

classmethod _allowed_child_types()

_str_helper()

property callee

property feature

exception annize.project.**FeatureUnavailableError**(*featurename*)

Bases: ModuleNotFoundError

Parameters **featurename** (*str*) –

class annize.project.**FileNode**(*path*)

Bases: *annize.project.Node*

Parameters **path** (*str*) –

classmethod _allowed_child_types()

_str_helper()

property path

exception annize.project.**MaterializerError**(*message*)

Bases: TypeError

Parameters **message** (*str*) –

class annize.project.**Node**

Bases: object

__get_description(*indent*, *with_children*, *multiline*)

Parameters

- **indent** (*int*) –
- **with_children** (*bool*) –

- **multiline** (*bool*) –

Return type str

classmethod _allowed_child_types()

Return type Iterable[Type[*annize.project.Node*]]

_str_helper()

append_child(*node*)

Parameters **node** (annize.project.Node) –

property children

clone(*with_children=True*)

Parameters **with_children** (*bool*) –

Return type *annize.project.Node*

get_description(**, with_children=True, multiline=True*)

Parameters

- **with_children** (*bool*) –
- **multiline** (*bool*) –

Return type str

insert_child(*i, node*)

Parameters

- **i** (*int*) –
- **node** (annize.project.Node) –

property parent

remove_child(*node*)

Parameters **node** (annize.project.Node) –

class annize.project.**OnFeatureUnavailableNode**

Bases: *annize.project.ScopableBlockNode*

class Action(*value*)

Bases: enum.Enum

An enumeration.

FAIL = 'fail'

SKIP_BLOCK = 'skip_block'

SKIP_NODE = 'skip_node'

_str_helper()

property do

property feature

exception annize.project.**ParserError**(*message*)

Bases: ValueError

Parsing error like bad input xml.

Parameters **message** (*str*) –

class annize.project.**ProjectDefinition**(**nodes*)

Bases: object

A data structure that holds an in-memory representation of the project description. It is not the in-memory representation of the live universe object state. It is like the file structure of the project description file, but not exactly the same. There is a parser and a generator for translating between a project description file and an ProjectDefinition.

Parameters **nodes** (FileNode) –

static from_files(*path*)

Parameters **path** (*str*) –

Return type *annize.project.ProjectDefinition*

property nodes

class annize.project.**ReferenceNode**

Bases: *annize.project.ArgumentNode*

class OnUnresolvableAction(*value*)

Bases: enum.Enum

An enumeration.

FAIL = 'fail'

SKIP = 'skip'

classmethod _allowed_child_types()

_str_helper()

property on_unresolvable

property reference_key

class annize.project.**ScalarValueNode**

Bases: *annize.project.ArgumentNode*

__shorten(*maxlen=100*)

Parameters

- **obj** (*Any*) –
- **maxlen** (*int*) –

Return type str

_str_helper()

property value

class annize.project.**ScopableBlockNode**

Bases: *annize.project.BlockNode*

class Scope(*value*)

Bases: enum.Enum

An enumeration.

BLOCK = 'block'

FILE = 'file'

PROJECT = 'project'

_str_helper()

property scope

exception annize.project.**UnresolvableReferenceError**(*refname*)

Bases: *annize.project.MaterializerError*

Parameters **refname** (*str*) –

Subpackages

annize.project.materializer package

Materializing of Annize projects into a working structure.

annize.project.materializer.**materialize**(*nodes*, **, *featureloader=None*)

Parameters

- **nodes** (*Iterable[*annize.project.FileNode*]*) –
- **featureloader** (*Optional[*annize.project.featureloader.FeatureLoader*]*) –

Return type List[Optional[Any]]

Subpackages

annize.project.materializer.behaviors package

class annize.project.materializer.behaviors.**Behavior**

Bases: abc.ABC

_abc_impl = <_abc_data object>

abstract node_context(*nodemat*)

Parameters **nodemat** (annize.project.materializer.core.NodeMaterialization) –

Return type AbstractContextManager

stop_materialization_here_and_try_again()

Return type None

Submodules

annize.project.materializer.behaviors.argument module

class annize.project.materializer.behaviors.argument.**ArgumentBehavior**(*callfct*, ***, *featureloader*)

Bases: *annize.project.materializer.behaviors.Behavior*

Parameters **featureloader** (annize.project.featureloader.FeatureLoader) –

_abc_impl = <_abc_data object>

node_context (*nodemat*)

annize.project.materializer.behaviors.basket module

class annize.project.materializer.behaviors.basket.**BasketBehavior**

Bases: *annize.project.materializer.behaviors.Behavior*

_abc_impl = <_abc_data object>

node_context (*nodemat*)

annize.project.materializer.behaviors.block module

class annize.project.materializer.behaviors.block.**BlockBehavior**

Bases: *annize.project.materializer.behaviors.Behavior*

_abc_impl = <_abc_data object>

node_context (*nodemat*)

annize.project.materializer.behaviors.featureunavailable module

class annize.project.materializer.behaviors.featureunavailable.**FeatureUnavailableBehavior**

Bases: *annize.project.materializer.behaviors.Behavior*

__context_catchexceptions (*nodemat*, *featureignorelist*)

__context_skipnode_featureignorelist (*node*)

_abc_impl = <_abc_data object>

node_context (*nodemat*)

annize.project.materializer.behaviors.reference module

class annize.project.materializer.behaviors.reference.**ReferenceBehavior** (*rootblock*)

Bases: *annize.project.materializer.behaviors.Behavior*

_abc_impl = <_abc_data object>

detect_retry_useful_node_context (*nodemat*)

node_context (*nodemat*)

resolver_node_context (*nodemat*)

retry_node_context (*nodemat*)

class annize.project.materializer.behaviors.reference.**SkipUnresolvableReferenceBehavior** (*roo*

Bases: *annize.project.materializer.behaviors.Behavior*

classmethod __node_is_hopeless (*node*, *rootnodemat*)

Parameters **node** (annize.project.Node) –

Return type bool

`__nodes_by_name`(*, *rootnode*)

Parameters

- **name** (*str*) –
- **rootnode** (`annize.project.Node`) –

Return type List[*annize.project.Node*]

`classmethod` **`__reference_is_hopeless`**(*refnode*, *, *rootnodemat*)

Parameters **refnode** (`annize.project.ReferenceNode`) –

Return type bool

`__unresolvable_references`()

Return type List[*annize.project.ReferenceNode*]

`_abc_impl = <_abc_data object>`

`node_context`(*nodemat*)

Submodules

annize.project.materializer.core module

`class` `annize.project.materializer.core.`**`NodeMaterialization`**(*materializer*, *node*, *store*)

Bases: `object`

Parameters

- **materializer** (`ProjectMaterializer`) –
- **node** (`annize.project.Node`) –
- **store** (*Dict*) –

`get_materialized_children`()

Return type Iterable[Optional[Any]]

`get_materialized_children_tuples`()

`property` **`has_result`**

`property` **`node`**

`property` **`result`**

`set_materialized_result`(*resultlist*)

`try_get_materialization_for_node`(*node*)

Parameters **node** (`annize.project.Node`) –

`class` `annize.project.materializer.core.`**`ProjectMaterializer`**(*node*, *, *behaviors*)

Bases: `object`

Parameters

- **node** (`annize.project.Node`) –

- **behaviors** (*Iterable[annize.project.materializer.behaviors.Behavior]*) –

__materialization_for_node(*node*, *store*)

Parameters

- **node** (annize.project.Node) –
- **store** (*Dict*) –

Return type *annize.project.materializer.core.NodeMaterialization*

__materialize(*node*, *store*)

Parameters

- **node** (annize.project.Node) –
- **store** (*Dict*) –

Return type List[Optional[Any]]

_materialize_hlp_childobjs(*node*, *store*)

Parameters

- **node** (annize.project.Node) –
- **store** (*Dict*) –

Return type List[Tuple[*annize.project.Node*, List[Optional[Any]]]]

get_materialized()

Return type List[Optional[Any]]

exception annize.project.materializer.core.**TryAgain**

Bases: BaseException

annize.project.materializer.preprocessors module

annize.project.materializer.preprocessors.**normalize_blockscopes**(*nodes*)

Parameters **nodes** (*Iterable[annize.project.Node]*) –

Return type Iterable[*annize.project.Node*]

annize.project.materializer.preprocessors.**resolve_appendtonodes**(*nodes*)

Parameters **nodes** (*Iterable[annize.project.Node]*) –

Return type Iterable[*annize.project.Node*]

Submodules

annize.project.featureloader module

Feature module loader.

class annize.project.featureloader.**DefaultFeatureLoader**

Bases: *annize.project.featureloader.FeatureLoader*

_FEATURES_NAMESPACE = 'annize.features'

_abc_impl = <_abc_data object>

load_feature (*name*)

class annize.project.featureloader.**FeatureLoader**

Bases: abc.ABC

_abc_impl = <_abc_data object>

abstract load_feature (*name*)

Parameters **name** (*str*) –

Return type Optional[Any]

annize.project.parser module

Reading, processing and writing of Annize projects.

class annize.project.parser.**_XmlParser**

Bases: object

ATTRIBUTE_COMMAND_END = '~'

ATTRIBUTE_COMMAND_START = '~'

class _Context

Bases: object

static _Context__nodeshort (*xnode*, *maxlen=100*)

Parameters **xnode** (*xml.etree.ElementTree.Element*) –

in_file (*fpath*)

in_node (*xnode*)

class _TagParts (*name*, *namespace=''*)

Bases: object

Parameters **namespace** (*str*) –

classmethod _XmlParser__interpret_attribute_string (*txt*)

Parameters **txt** (*str*) –

Return type Tuple[bool, str]

classmethod _XmlParser__parse_attrib (*key*, *value*)

Parameters

- **key** (*str*) –
- **value** (*str*) –

Return type *annize.project.Node*

_XmlParser__parse_child(*node*, *xnode*)

Parameters

- **node** (annize.project.Node) –
- **xnode** (*xml.etree.ElementTree.Element*) –

Return type Tuple[*annize.project.Node*, xml.etree.ElementTree.Element]

_XmlParser__parse_children(*node*, *xparent*)

Parameters

- **node** (annize.project.Node) –
- **xparent** (*xml.etree.ElementTree.Element*) –

_XmlParser__parse_tag(*node*, *argname*, *callname*, *feature*, *xnode*)

Parameters **node** (annize.project.Node) –

classmethod escape_attribute_string(*txt*)

Parameters **txt** (*str*) –

Return type str

parse_file(*fpath*)

Parameters **fpath** (*str*) –

Return type *annize.project.FileNode*

annize.project.parser.**parse**(*path*)

Parameters **path** (*str*) –

Return type List[*annize.project.FileNode*]

annize.userfeedback package

class annize.userfeedback.**NullUserFeedbackController**

Bases: object

choice_dialog(**_*)

input_dialog(**_*)

message_dialog(**_*)

exception annize.userfeedback.**UnsatisfiableUserFeedbackAttemptError**

Bases: RuntimeError

class annize.userfeedback.**UserFeedbackController**

Bases: object

choice_dialog(*question*, *choices*, *config_key*)

Parameters

- **question** (*str*) –
- **choices** (*List[str]*) –
- **config_key** (*Optional[str]*) –

Return type Optional[int]

input_dialog(*question*, *suggested_answer*, *config_key*)

Parameters

- **question** (*str*) –
- **suggested_answer** (*str*) –
- **config_key** (*Optional[str]*) –

Return type Optional[str]

message_dialog(*message*, *answers*, *config_key*)

Parameters

- **message** (*str*) –
- **answers** (*List[str]*) –
- **config_key** (*Optional[str]*) –

Return type int

annize.userfeedback.**_add_controller_to_context**(**, *controller*, *context*, *priority_index=0*)

Parameters

- **controller** (annize.userfeedback.UserFeedbackController) –
- **context** (annize.context.Context) –
- **priority_index** (*int*) –

Return type None

annize.userfeedback.**_controllers_for_context**(*context*)

Parameters **context** (annize.context.Context) –

Return type List[*annize.userfeedback.UserFeedbackController*]

annize.userfeedback.**_controllers_tuples_for_context**(*context*)

Parameters **context** (annize.context.Context) –

Return type List[Tuple[int, *annize.userfeedback.UserFeedbackController*]]

annize.userfeedback.**choice_dialog**(*message*, *choices*, *, *config_key=None*)

Parameters

- **message** (*Union[str,* annize.i18n.TrStr*]*) –
- **choices** (*Iterable[Union[str,* annize.i18n.TrStr*]]*) –
- **config_key** (*Optional[str]*) –

Return type Optional[int]

annize.userfeedback.**input_dialog**(*message*, *, *suggested_answer*, *config_key=None*)

Parameters

- **message** (*Union[str,* annize.i18n.TrStr*]*) –
- **suggested_answer** (*Union[str,* annize.i18n.TrStr*]*) –
- **config_key** (*Optional[str]*) –

Return type Optional[str]

`annize.userfeedback.`**`message_dialog`**(*message*, *answers*, ***, *config_key=None*)

Parameters

- **`message`** (*`Union[str, annize.i18n.TrStr]`*) –
- **`answers`** (*`Iterable[Union[str, annize.i18n.TrStr]]`*) –
- **`config_key`** (*`Optional[str]`*) –

Return type int

Submodules

annize.userfeedback.static module

`class` `annize.userfeedback.static.`**`StaticUserFeedbackController`**(*answers*)

Bases: *`annize.userfeedback.UserFeedbackController`*

Parameters **`answers`** (*`Dict[str, Any]`*) –

`__get_answer`(*config_key*)

Parameters **`config_key`** (*`str`*) –

Return type Any

`add_answer`(*config_key*, *value*)

Parameters

- **`config_key`** (*`str`*) –
- **`value`** (*`Any`*) –

Return type None

`choice_dialog`(*question*, *choices*, *config_key*)

`input_dialog`(*question*, *suggested_answer*, *config_key*)

`message_dialog`(*message*, *answers*, *config_key*)

annize.userfeedback.tty module

`class` `annize.userfeedback.tty.`**`TtyUserFeedbackController`**

Bases: *`annize.userfeedback.UserFeedbackController`*

`__action_line`(*num*, *text*)

Parameters

- **`num`** (*`Any`*) –
- **`text`** (*`str`*) –

Return type str

`__dialog`(*message*, *config_key*, *actionstext*)

Parameters

- **`message`** (*`str`*) –

- **config_key** (*str*) –
- **actionstext** (*str*) –

Return type str

__dialog_frame_actions (*text*)

Parameters **text** (*str*) –

Return type str

__dialog_frame_configkey (*config_key*)

Parameters **config_key** (*str*) –

Return type str

__dialog_frame_message (*message*)

Parameters **message** (*str*) –

Return type str

choice_dialog (*question*, *choices*, *config_key*)

input_dialog (*question*, *suggested_answer*, *config_key*)

message_dialog (*message*, *answers*, *config_key*)

6.1.2 Submodules

6.1.3 annize.annize_cli module

class annize.annize_cli.**Cli**

Bases: object

classmethod __answers_from_json_files (*destination*, *with_answers_from_json_files*)

Parameters

- **destination** (*Dict*) –
- **with_answers_from_json_files** (*Iterable[str]*) –

classmethod __answers_from_json_strings (*destination*, *with_answers_from_json_strings*)

Parameters

- **destination** (*Dict*) –
- **with_answers_from_json_strings** (*Iterable[str]*) –

classmethod __answers_from_single_answers (*destination*, *with_answers*)

Parameters

- **destination** (*Dict*) –
- **with_answers** (*Iterable[Tuple[str, str]]*) –

classmethod do (*project*, *taskname*, *with_answers_from_json_file*, *with_answers_from_json_string*, *with_answer*, ***_*)

Parameters

- **project** (*str*) –

- **taskname** (*str*) –
- **with_answers_from_json_file** (*Iterable[str]*) –
- **with_answers_from_json_string** (*Iterable[str]*) –
- **with_answer** (*Iterable[Tuple[str, str]]*) –

project_default = '/home/pino/projects/annize/src/annize'

annize.annize_cli.**get_parser**()

Return type argparse.ArgumentParser

annize.annize_cli.**main**()

6.1.4 annize.engine module

class annize.engine.**LoadedProject**(*context*, *null_culture*)

Bases: object

Parameters

- **context** (annize.context.Context) –
- **null_culture** (annize.i18n.Culture) –

property context

annize.engine.**_prepare_context**(*context*, ***, *annize_config_rootpath*)

Parameters

- **context** (annize.context.Context) –
- **annize_config_rootpath** (*str*) –

Return type None

annize.engine.**find_project_annize_config_root_file**(*project_path*)

Parameters **project_path** (*str*) –

Return type Optional[str]

annize.engine.**guess_project_root_directory**(*annize_config_rootpath*)

Parameters **annize_config_rootpath** (*str*) –

Return type str

annize.engine.**load_project**(*project_path*)

Parameters **project_path** (*str*) –

Return type *annize.engine.LoadedProject*

annize.engine.**load_project_definition**(*projectdefinition*, ***, *annize_config_rootpath*)

Parameters

- **projectdefinition** (annize.project.ProjectDefinition) –
- **annize_config_rootpath** (*str*) –

Return type *annize.engine.LoadedProject*

PYTHON MODULE INDEX

INDEX

Symbols