
lawwenda - Plugin API Documentation

Release 0.2.104

Josef Hahn

Mar 20, 2021

CONTENTS

1	API reference	3
1.1	lawwenda package	3
	Python Module Index	33
	Index	35

Lawwenda has a plugin interface which allows to add functionality to the core from outside.

TODO not yet available

API REFERENCE

1.1 lawwenda package

1.1.1 Subpackages

lawwenda._aux package

Submodules

lawwenda._aux.projectinformation module

Module contents

lawwenda.preview package

Submodules

lawwenda.preview.commonimages module

```
class lawwenda.preview.commonimages.CommonImagesPreviewer
    Bases: lawwenda.preview.Previewer
    _abc_impl = <_abc_data object>
    is_previewable(iar)
    is_thumbnailable(iar)
    preview_html(node)
    thumbnail(node)
```

lawwenda.preview.commonvideos module

class lawwenda.preview.commonvideos.CommonVideosPreviewer

Bases: *lawwenda.preview.Previewer*

__ffmpeg_thumb (*stdin=- 3*)

Parameters

- **inputpath** (*str*) –
- **stdin** (*Any*) –

Return type bytes

_abc_impl = <_abc_data object>

is_previewable (*iar*)

is_thumbnailable (*iar*)

preview_html (*node*)

thumbnail (*node*)

Module contents

class lawwenda.preview.IsAbleRequest (*name: str, mimetype: str*)

Bases: object

Parameters

- **name** (*str*) –
- **mimetype** (*str*) –

Return type None

mimetype: str

name: str

class lawwenda.preview.Previewer

Bases: abc.ABC

_abc_impl = <_abc_data object>

_image_scale_to_thumbnail_size (*imgdata*)

Parameters *imgdata* (*BinaryIO*) –

Return type *_io.BytesIO*

abstract is_previewable (*iar*)

Parameters *iar* (*lawwenda.preview.IsAbleRequest*) –

Return type bool

abstract is_thumbnailable (*iar*)

Parameters *iar* (*lawwenda.preview.IsAbleRequest*) –

Return type bool

abstract preview_html (*node*)

Parameters `node` (`lawwenda.fs.Filesystem.Node`) –

Return type `str`

`thumbheight = 300`

`abstract thumbnail` (`node`)

Parameters `node` (`lawwenda.fs.Filesystem.Node`) –

Return type `bytes`

`thumbsize = (300, 300)`

`thumbwidth = 300`

lawwenda.search package

Module contents

class `lawwenda.search.ByGeoSearch` (`position`, `radius`)

Bases: `lawwenda.search.Search`

Parameters

- `position` (`str`) –
- `radius` (`str`) –

`__decide` (`node`)

Parameters `node` (`FilesystemNode`) –

Return type `bool`

`query` (`node`)

class `lawwenda.search.ByNameSearch` (`term`)

Bases: `lawwenda.search.Search`

Parameters `term` (`str`) –

`__decide` (`node`)

Parameters `node` (`FilesystemNode`) –

Return type `bool`

`query` (`node`)

class `lawwenda.search.ByTagSearch` (`term`)

Bases: `lawwenda.search.Search`

Parameters `term` (`str`) –

`__decide` (`node`)

Parameters `node` (`FilesystemNode`) –

Return type `bool`

`query` (`node`)

class `lawwenda.search.DeeplySearch` (`term`)

Bases: `lawwenda.search.Search`

Parameters `term` (`str`) –

__decide_unindexed (*node*)

Parameters **node** (*FileSystemNode*) –

Return type bool

__decide_unindexed_by_content (*node*)

Parameters **node** (*FileSystemNode*) –

Return type bool

__decide_unindexed_by_metadata (*node*)

Parameters **node** (*FileSystemNode*) –

Return type bool

__query_indexed (*idxdata*)

Parameters

- **node** (*FileSystemNode*) –
- **idxdata** (*object*) –

Return type *FileSystemNodes*

__query_unindexed ()

Parameters **node** (*FileSystemNode*) –

Return type *FileSystemNodes*

__try_get_index (*node*)

Parameters **node** (*FileSystemNode*) –

Return type *object*

query (*node*)

class lawwenda.search.**Search**

Bases: *object*

query (*node*)

Parameters **node** (*FileSystemNode*) –

Return type *FileSystemNodes*

lawwenda.search.**__query_by_tree_traversal** (*node*, *decidefct*)

Parameters

- **node** (*FileSystemNode*) –
- **decidefct** (*Callable[[FileSystemNode], bool]*) –

Return type *FileSystemNodes*

lawwenda.search.**__termstring_to_termlist** (*term*)

Parameters **term** (*str*) –

Return type *List[str]*

lawwenda.search.**create_search** (*mode*, ***kwargs*)

Parameters **mode** (*str*) –

Return type *lawwenda.search.Search*

lawwenda.sharehttpapp package

Submodules

lawwenda.sharehttpapp.davprop module

class lawwenda.sharehttpapp.davprop.**CommentDavProp**

Bases: *lawwenda.sharehttpapp.davprop.DavProp*

get_for_node (*fsnode*)

class lawwenda.sharehttpapp.davprop.**DavProp** (*davname*, *, *is_writable*)

Bases: object

Parameters

- **davname** (*str*) –
- **is_writable** (*bool*) –

property **davname**

Return type *str*

Return type *str*

Return type *str*

Return type *str*

get_for_node (*fsnode*)

Parameters **fsnode** (*lawwenda.fs.Filesystem.Node*) –

Return type *Optional[str]*

property **is_writable**

Return type *bool*

Return type *bool*

Return type *bool*

Return type *bool*

class lawwenda.sharehttpapp.davprop.**GeoDavProp**

Bases: *lawwenda.sharehttpapp.davprop.DavProp*

get_for_node (*fsnode*)

class lawwenda.sharehttpapp.davprop.**MtimeDavProp**

Bases: *lawwenda.sharehttpapp.davprop.DavProp*

get_for_node (*fsnode*)

class lawwenda.sharehttpapp.davprop.**RatingDavProp**

Bases: *lawwenda.sharehttpapp.davprop.DavProp*

get_for_node (*fsnode*)

class lawwenda.sharehttpapp.davprop.**ResourceTypeDavProp**

Bases: *lawwenda.sharehttpapp.davprop.DavProp*

```
    get_for_node (fsnode)
class lawwenda.sharehttpapp.davprop.SizeDavProp
    Bases: lawwenda.sharehttpapp.davprop.DavProp
    get_for_node (fsnode)
class lawwenda.sharehttpapp.davprop.TagsDavProp
    Bases: lawwenda.sharehttpapp.davprop.DavProp
    get_for_node (fsnode)
lawwenda.sharehttpapp.davprop.get_prop_by_davname (davname)
    Parameters davname (str) –
    Return type Optional[lawwenda.sharehttpapp.davprop.DavProp]
lawwenda.sharehttpapp.davprop.register_prop (prop)
    Parameters prop (Type [DavProp]) –
    Return type Type[lawwenda.sharehttpapp.davprop.DavProp]
```

Module contents

```
class lawwenda.sharehttpapp.ShareHttpApp (filesystem)
    Bases: object
    TODO.
    Parameters filesystem (lawwenda.fs.Filesystem) –
    property __allowed_methods
        Return type t.List[str]
        Return type t.List[str]
        Return type t.List[str]
        Return type t.List[str]
    _method_copy (request, fsnode, *, xfermethod='copy_to')
        Parameters
            • request (lawwenda.comm.Request) –
            • fsnode (lawwenda.fs.Filesystem.Node) –
            • xfermethod (str) –
    _method_delete (request, fsnode)
        Parameters
            • request (lawwenda.comm.Request) –
            • fsnode (lawwenda.fs.Filesystem.Node) –
    _method_get (request, fsnode, is_head_request=False)
        Parameters
            • request (lawwenda.comm.Request) –
            • fsnode (lawwenda.fs.Filesystem.Node) –
```

- **is_head_request** (*bool*) –

_method_head (*request, fsnode*)

Parameters

- **request** (*lawwenda.comm.Request*) –
- **fsnode** (*lawwenda.fs.Filesystem.Node*) –

_method_mkcol (*request, fsnode*)

Parameters

- **request** (*lawwenda.comm.Request*) –
- **fsnode** (*lawwenda.fs.Filesystem.Node*) –

_method_move (*request, fsnode*)

Parameters

- **request** (*lawwenda.comm.Request*) –
- **fsnode** (*lawwenda.fs.Filesystem.Node*) –

_method_options (*request, fsnode*)

Parameters

- **request** (*lawwenda.comm.Request*) –
- **fsnode** (*lawwenda.fs.Filesystem.Node*) –

_method_propfind (*request, fsnode*)

Parameters

- **request** (*lawwenda.comm.Request*) –
- **fsnode** (*lawwenda.fs.Filesystem.Node*) –

_method_proppatch (*request, fsnode*)

Parameters

- **request** (*lawwenda.comm.Request*) –
- **fsnode** (*lawwenda.fs.Filesystem.Node*) –

_method_put (*request, fsnode*)

Parameters

- **request** (*lawwenda.comm.Request*) –
- **fsnode** (*lawwenda.fs.Filesystem.Node*) –

1.1.2 Submodules

1.1.3 lawwenda.comm module

class lawwenda.comm.**JsonedResponse** (*data, **kwargs*)

Bases: `werkzeug.wrappers.response.Response`

Parameters *data* (*Optional[Any]*) –

class lawwenda.comm.**Request** (*environ, populate_request=True, shallow=False*)

Bases: `werkzeug.wrappers.request.Request`, `werkzeug.wrappers.json.JSONMixin`

1.1.4 lawwenda.datafiles module

lawwenda.datafiles.**find_data_file** (*fname, searchdirs=None*)

Parameters

- **fname** (*str*) –
- **searchdirs** (*Optional[Iterable[str]]*) –

Return type `Optional[str]`

1.1.5 lawwenda.devserver module

class lawwenda.devserver.**__DevServerInfo** (*svr, svrthread*)

Bases: `object`

shutdown ()

Return type `None`

property url

Return type `str`

Return type `str`

Return type `str`

Return type `str`

wait_stopped ()

Return type `None`

class lawwenda.devserver.**__DevServerThread** (*svr*)

Bases: `threading.Thread`

This constructor should always be called with keyword arguments. Arguments are:

group should be `None`; reserved for future extension when a `ThreadGroup` class is implemented.

target is the callable object to be invoked by the `run()` method. Defaults to `None`, meaning nothing is called.

name is the thread name. By default, a unique name is constructed of the form “Thread-N” where N is a small decimal number.

args is the argument tuple for the target invocation. Defaults to `()`.

kwargs is a dictionary of keyword arguments for the target invocation. Defaults to `{}`.

If a subclass overrides the constructor, it must make sure to invoke the base class constructor (`Thread.__init__()`) before doing anything else to the thread.

run()

Method representing the thread's activity.

You may override this method in a subclass. The standard `run()` method invokes the callable object passed to the object's constructor as the target argument, if any, with sequential and keyword arguments taken from the `args` and `kwargs` arguments, respectively.

`lawwenda.devserver.get_running_dev_servers()`

Return type `Iterable[lawwenda.devserver._DevServerInfo]`

`lawwenda.devserver.run_dev_server(cfg)`

Parameters `cfg` (`lawwenda.Configuration`) –

Return type `lawwenda.devserver._DevServerInfo`

1.1.6 lawwenda.fmapp module

class `lawwenda.fmapp.FmApp` (*share*)

Bases: `object`

Parameters `share` (`lawwenda.Share`) –

`__auth` (*username*, *password*)

Parameters

- `username` (*str*) –
- `password` (*str*) –

Return type `bool`

`__dispatch_request` (*request*)

`__dispatch_request_internals` (*request*)

`__dispatch_request_normal` (*request*)

`__ensure_authed` (*request*)

`__on_api_copymove` (*request*, *action*)

`__render_template` (*template*, *, *html_head_inner*="", *path*="", *headonly*=False, *url_internals_name*='~__lawwenda__int~', ***kwargs*)

Parameters

- `template` (*str*) –
- `html_head_inner` (*str*) –
- `path` (*str*) –
- `headonly` (*bool*) –
- `url_internals_name` (*str*) –

Return type `werkzeug.wrappers.response.Response`

`__render_template_text` (*template*, *, *csrftoken*, *html_head_inner*, ***kwargs*)

Parameters

- **template** (*str*) –
- **csrftoken** (*str*) –
- **html_head_inner** (*str*) –

Return type *str*

__render_template_text_raw (*template*, ***kwargs*)

Parameters **template** (*str*) –

Return type *str*

_on_api_copy (*request*)

_on_api_delete (*request*)

_on_api_details (*request*)

_on_api_dir (*request*)

_on_api_known_tags (*_*)

_on_api_mkdir (*request*)

_on_api_move (*request*)

_on_api_rename (*request*)

_on_api_set_comment (*request*)

_on_api_set_geo (*request*)

_on_api_set_rating (*request*)

_on_api_tag_entries (*request*)

_on_api_thumbnail (*request*)

_on_api_untag_entries (*request*)

_on_api_upload (*request*)

_on_api_zip (*request*)

_on_api_zip_download (*request*, *zipid*)

_on_help (*_*)

_on_static (*_*, *filepath*)

url_internals_name = *'~__lawwenda__int~'*

class lawwenda.fmapp._RenderTemplateValue (*s*)

Bases: *object*

property *unescaped*

class lawwenda.fmapp._TempZips

Bases: *object*

class _TempZip (*nodes*)

Bases: *object*

Parameters **nodes** (*List* [*lawwenda.fs.Filesystem.Node*]) –

_TempZip__is_executable (*node*)

Parameters **node** (*lawwenda.fs.Filesystem.Node*) –

Return type *bool*


```

    _TempZip__putnode(node, zipf)
        Parameters
            • node (lawwenda.fs.Filesystem.Node) –
            • zipf (zipfile.ZipFile) –
        Return type None

    property bytes

    classmethod _TempZips__cleanup_loop()

    _cleanup_thread = None

    _lock = <unlocked _thread.lock object>

    _zips = {}

    classmethod create_tempzip(owner, nodes)
        Parameters
            • owner (object) –
            • nodes (List[lawwenda.fs.Filesystem.Node]) –
        Return type str

    classmethod get_tempzip(owner, zipid)
        Parameters
            • owner (object) –
            • zipid (str) –
        Return type Optional[bytes]

```

1.1.7 lawwenda.fs module

```

class lawwenda.fs.AbstractFilesystemDecorator(inner)
    Bases: lawwenda.fs.Filesystem
        Parameters inner (lawwenda.fs.Filesystem) –
        _eval_predicate(pred, node)

exception lawwenda.fs.CircularTraversalError
    Bases: OSError

class lawwenda.fs.ExcludeNodesFilesystemAdapter(inner, pred)
    Bases: lawwenda.fs.AbstractFilesystemDecorator
        Parameters
            • inner (lawwenda.fs.Filesystem) –
            • pred (Callable[[Filesystem.Node], bool]) –
        child_nodes(handle)
            Parameters handle (lawwenda.fs.Filesystem.ReadHandle) –
            Return type List[lawwenda.fs.Filesystem.Node]
        get_readhandle(node)
        get_writehandle(node)

```

```
class lawwenda.fs.Filesystem
    Bases: object

    class Node(path, *, filesystem)
        Bases: object

        Parameters

        • path(str) –
        • filesystem(Filesystem) –

    property _filesystem
        Return type Filesystem
        Return type Filesystem
        Return type Filesystem
        Return type Filesystem

    add_tag(tag)
        Parameters tag(str) –
        Return type None

    property basics_as_dict

    child_by_name(name)
        Parameters name(str) –
        Return type lawwenda.fs.Filesystem.Node

    property child_nodes
        Return type t.List[“Filesystem.Node”]
        Return type t.List[“Filesystem.Node”]
        Return type t.List[“Filesystem.Node”]
        Return type t.List[“Filesystem.Node”]

    property comment
        Return type str
        Return type str
        Return type str
        Return type str

    copy_to(newpath)
        Parameters newpath(str) –
        Return type None

    delete()
        Return type None

    property dirpath
        Return type str
        Return type str
        Return type str
        Return type str

    property exists
        Return type bool
        Return type bool
        Return type bool
        Return type bool

    property full_as_dict

    property geo
```

Return type str

Return type str

Return type str

Return type str

property geo_obj

Return type pimetadatainterpreter.GeoLocation

Return type pimetadatainterpreter.GeoLocation

Return type pimetadatainterpreter.GeoLocation

Return type pimetadatainterpreter.GeoLocation

property has_preview

Return type bool

Return type bool

Return type bool

Return type bool

property has_thumbnail

Return type bool

Return type bool

Return type bool

Return type bool

property icon_name

Return type t.Optional[str]

Return type t.Optional[str]

Return type t.Optional[str]

Return type t.Optional[str]

property is_dir

Return type bool

Return type bool

Return type bool

Return type bool

property is_file

Return type bool

Return type bool

Return type bool

Return type bool

property is_hidden

Return type bool

Return type bool

Return type bool

Return type bool

property is_link

Return type bool

Return type bool

Return type bool

Return type bool

property is_writable

Return type bool

Return type bool

Return type bool

Return type bool

property mimetype
Return type str
Return type str
Return type str
Return type str

mkdir()
Return type None

move_to(newpath)
Parameters **newpath** (*str*) –
Return type None

property mtime
Return type datetime.datetime
Return type datetime.datetime
Return type datetime.datetime
Return type datetime.datetime

property mtime_ts
Return type float
Return type float
Return type float
Return type float

property name
Return type str
Return type str
Return type str
Return type str

property parent_node
Return type *Filesystem.Node*
Return type *Filesystem.Node*
Return type *Filesystem.Node*
Return type *Filesystem.Node*

property path
Return type str
Return type str
Return type str
Return type str

property preview_html
Return type str
Return type str
Return type str
Return type str

property rating
Return type int
Return type int
Return type int
Return type int

read_file()
Return type BinaryIO

remove_tag(tag)

```

    Parameters tag (str) –
    Return type None

set_comment (comment)
    Parameters comment (str) –
    Return type None

set_geo (geo)
    Parameters geo (str) –
    Return type None

set_rating (rating)
    Parameters rating (int) –
    Return type None

property size
    Return type int
    Return type int
    Return type int
    Return type int

property tags
    Return type t.List[str]
    Return type t.List[str]
    Return type t.List[str]
    Return type t.List[str]

property tagstring
    Return type str
    Return type str
    Return type str
    Return type str

thumbnail ()
    Return type bytes

traverse_dir (*, raise_on_circle, param_path="")
    Parameters
      • raise_on_circle (bool) –
      • param_path (str) –
    Return type Iterable[Tuple[lawwenda.fs.Filesystem.Node, str]]

try_get_fullpath (*, writable)
    Parameters writable (bool) –
    Return type Optional[str]

write_file (content)
    Parameters content (Union[bytes, BinaryIO]) –
    Return type None

```

```
class ReadHandle (node)
```

Bases: object

Read and write handles are just a stupid container that hold one *Filesystem.Node*.

This looks stupid at first, because you could just use this node directly instead. The added value of handles are a central mechanism for access control, which would be a bit less obvious and more scattered in code without this indirection.

Of course it cannot avoid a way around it in code. An attacker that can change the code has won anyway. It just simplifies writing correct code that hopefully does not provide ways around it for the client.

See `Filesystem.get_readhandle()` and `Filesystem.get_writehandle()`.

Parameters `node` (`Filesystem.Node`) –

class `WriteHandle` (`node`)

Bases: `lawwenda.fs.Filesystem.ReadHandle`

See `Filesystem.ReadHandle`.

Parameters `node` (`Filesystem.Node`) –

add_tag (`handle`, `tag`)

Parameters

- **handle** (`lawwenda.fs.Filesystem.WriteHandle`) –
- **tag** (`str`) –

Return type `None`

child_nodes (`handle`)

Parameters `handle` (`lawwenda.fs.Filesystem.ReadHandle`) –

Return type `List[lawwenda.fs.Filesystem.Node]`

comment (`handle`)

Parameters `handle` (`lawwenda.fs.Filesystem.ReadHandle`) –

Return type `str`

copy_to (`srchandle`, `desthandle`)

Parameters

- **srchandle** (`lawwenda.fs.Filesystem.ReadHandle`) –
- **desthandle** (`lawwenda.fs.Filesystem.WriteHandle`) –

Return type `None`

delete (`handle`)

Parameters `handle` (`lawwenda.fs.Filesystem.WriteHandle`) –

Return type `None`

exists (`handle`)

Parameters `handle` (`lawwenda.fs.Filesystem.ReadHandle`) –

Return type `bool`

geo (`handle`)

Parameters `handle` (`lawwenda.fs.Filesystem.ReadHandle`) –

Return type `lawwenda._aux.PiMetadataInterpreter.pimetadainterpreter.GeoLocation`

get_readhandle (`node`)

Parameters `node` (`lawwenda.fs.Filesystem.Node`) –

Return type `lawwenda.fs.Filesystem.ReadHandle`

get_writehandle (`node`)

Parameters `node` (`lawwenda.fs.Filesystem.Node`) –

Return type *lawwenda.fs.Filesystem.WriteHandle*

has_preview (*handle*)

Parameters **handle** (*lawwenda.fs.Filesystem.ReadHandle*) –

Return type bool

has_thumbnail (*handle*)

Parameters **handle** (*lawwenda.fs.Filesystem.ReadHandle*) –

Return type bool

is_dir (*handle*)

Parameters **handle** (*lawwenda.fs.Filesystem.ReadHandle*) –

Return type bool

is_file (*handle*)

Parameters **handle** (*lawwenda.fs.Filesystem.ReadHandle*) –

Return type bool

is_hidden (*handle*)

Parameters **handle** (*lawwenda.fs.Filesystem.ReadHandle*) –

Return type bool

is_link (*handle*)

Parameters **handle** (*lawwenda.fs.Filesystem.ReadHandle*) –

Return type bool

known_tags ()

Return type List[str]

mimetype (*handle*)

Parameters **handle** (*lawwenda.fs.Filesystem.ReadHandle*) –

Return type str

mkdir (*handle*)

Parameters **handle** (*lawwenda.fs.Filesystem.WriteHandle*) –

Return type None

move_to (*srchandle, desthandle*)

Parameters

- **srchandle** (*lawwenda.fs.Filesystem.WriteHandle*) –
- **desthandle** (*lawwenda.fs.Filesystem.WriteHandle*) –

Return type None

mtime (*handle*)

Parameters **handle** (*lawwenda.fs.Filesystem.ReadHandle*) –

Return type datetime.datetime

node_by_path (*path*)

Parameters `path` (*str*) –

Return type `lawwenda.fs.Filesystem.Node`

preview_html (*handle*)

Parameters `handle` (`lawwenda.fs.Filesystem.ReadHandle`) –

Return type `str`

rating (*handle*)

Parameters `handle` (`lawwenda.fs.Filesystem.ReadHandle`) –

Return type `int`

read_file (*handle*)

Parameters `handle` (`lawwenda.fs.Filesystem.ReadHandle`) –

Return type `BinaryIO`

remove_tag (*handle*, *tag*)

Parameters

- `handle` (`lawwenda.fs.Filesystem.WriteHandle`) –
- `tag` (*str*) –

Return type `None`

property rootnode

Return type `Filesystem.Node`

Return type `Filesystem.Node`

Return type `Filesystem.Node`

Return type `Filesystem.Node`

static sanitize_abspath (*path*)

Sanitizes slashes in a path and returns a path in the form */foo/bar*. Precisely:

- with a slash in the beginning
- without a slash at the end (exception: root path)
- without double slashes
- with `..` and `.` resolved
- root path: `/`

Parameters `path` (*str*) – The input path.

Return type `str`

set_comment (*handle*, *comment*)

Parameters

- `handle` (`lawwenda.fs.Filesystem.WriteHandle`) –
- `comment` (*str*) –

Return type `None`

set_geo (*handle*, *geo*)

Parameters

- **handle** (`lawwenda.fs.Filesystem.WriteHandle`) –
- **geo** (`str`) –

Return type `None`**set_rating** (`handle`, `rating`)**Parameters**

- **handle** (`lawwenda.fs.Filesystem.WriteHandle`) –
- **rating** (`int`) –

Return type `None`**size** (`handle`)**Parameters** **handle** (`lawwenda.fs.Filesystem.ReadHandle`) –**Return type** `int`**tags** (`handle`)**Parameters** **handle** (`lawwenda.fs.Filesystem.ReadHandle`) –**Return type** `List[str]`**thumbnail** (`handle`)**Parameters** **handle** (`lawwenda.fs.Filesystem.ReadHandle`) –**Return type** `bytes`**try_get_fullpath** (`handle`, `*`, `writable`)**Parameters**

- **handle** (`Union[Filesystem.ReadHandle, Filesystem.WriteHandle]`) –
- **writable** (`bool`) –

Return type `Optional[str]`**write_file** (`handle`, `content`)**Parameters**

- **handle** (`lawwenda.fs.Filesystem.WriteHandle`) –
- **content** (`Union[bytes, BinaryIO]`) –

Return type `None`**class** `lawwenda.fs.HideNodesFilesystemAdapter` (`inner`, `pred`)Bases: `lawwenda.fs.AbstractFilesystemDecorator`**Parameters**

- **inner** (`lawwenda.fs.Filesystem`) –
- **pred** (`Callable[[Filesystem.Node], bool]`) –

is_hidden (`handle`)**class** `lawwenda.fs.LocalFilesystem` (`rootpath`)Bases: `lawwenda.fs.Filesystem`

Parameters `rootpath` (*str*) –
`__path_to_fullpath` (*path*)

Parameters `path` (*str*) –

Return type `str`

`add_tag` (*handle*, *tag*)

`child_nodes` (*handle*)

`comment` (*handle*)

`copy_to` (*srchandle*, *desthandle*)

`delete` (*handle*)

`exists` (*handle*)

`geo` (*handle*)

`get_readhandle` (*node*)

`get_writehandle` (*node*)

`has_preview` (*handle*)

`has_thumbnail` (*handle*)

`is_dir` (*handle*)

`is_file` (*handle*)

`is_hidden` (*handle*)

`is_link` (*handle*)

`known_tags` ()

`mimetype` (*handle*)

`mkdir` (*handle*)

`move_to` (*srchandle*, *desthandle*)

`mtime` (*handle*)

`preview_html` (*handle*)

`rating` (*handle*)

`read_file` (*handle*)

`remove_tag` (*handle*, *tag*)

`set_comment` (*handle*, *comment*)

`set_geo` (*handle*, *geo*)

`set_rating` (*handle*, *rating*)

`size` (*handle*)

`tags` (*handle*)

`thumbnail` (*handle*)

`try_get_fullpath` (*handle*, *, *writable*)

`write_file` (*handle*, *content*)

```
class lawwenda.fs.ReadOnlyFileSystemAdapter(inner)  
    Bases: lawwenda.fs.AbstractFileSystemDecorator
```

```
        Parameters inner (lawwenda.fs.FileSystem) –  
        get_writehandle (node)
```

```
class lawwenda.fs._PredicateFactory  
    Bases: object
```

```
        static excludehidden (node)  
        static false (node)  
        static p_descending (pred)  
        static p_not (pred)  
        static p_or (*preds)  
        static p_regexp (restr)  
        static p_tag (tag)
```

```
lawwenda.fs.create_filesystem(rootpath, *, readonly, hide_by_patterns=(), hide_by_tags=(),  
                             include_by_patterns=None, include_by_tags=None,  
                             exclude_by_patterns=(), exclude_by_tags=(), ex-  
                             clude_hidden=False)
```

Parameters

- **rootpath** (*str*) –
- **readonly** (*bool*) –
- **hide_by_patterns** (*Iterable[str]*) –
- **hide_by_tags** (*Iterable[str]*) –
- **include_by_patterns** (*Optional[Iterable[str]]*) –
- **include_by_tags** (*Optional[Iterable[str]]*) –
- **exclude_by_patterns** (*Iterable[str]*) –
- **exclude_by_tags** (*Iterable[str]*) –
- **exclude_hidden** (*bool*) –

Return type *lawwenda.fs.FileSystem*

1.1.8 lawwenda.lawwenda_cli module

```
lawwenda.lawwenda_cli._fmt_cmd(txt)
```

Parameters *txt* (*str*) –

Return type *str*

```
lawwenda.lawwenda_cli._quickstart(cfgpath)
```

Parameters *cfgpath* (*str*) –

Return type *str*

```
lawwenda.lawwenda_cli.add_share(*, cfg, name, path, title, active_until, hide_by_pattern,  
                                hide_by_tag, include_by_pattern, include_by_tag, ex-  
                                clude_by_pattern, exclude_by_tag, exclude_hidden, readonly)
```

Parameters

- **cfg** (`lawwenda.Configuration`) –
- **name** (`str`) –
- **path** (`str`) –
- **title** (`str`) –
- **active_until** (`str`) –
- **hide_by_pattern** (`List[str]`) –
- **hide_by_tag** (`List[str]`) –
- **include_by_pattern** (`List[str]`) –
- **include_by_tag** (`List[str]`) –
- **exclude_by_pattern** (`List[str]`) –
- **exclude_by_tag** (`List[str]`) –
- **exclude_hidden** (`bool`) –
- **readonly** (`bool`) –

Return type None`lawwenda.lawwenda_cli.console(*, cfg)`**Parameters** **cfg** (`lawwenda.Configuration`) –**Return type** None`lawwenda.lawwenda_cli.describe_share(*, cfg, name)`**Parameters**

- **cfg** (`lawwenda.Configuration`) –
- **name** (`str`) –

Return type str`lawwenda.lawwenda_cli.generate_wsgi(*, cfg)`**Parameters** **cfg** (`lawwenda.Configuration`) –**Return type** str`lawwenda.lawwenda_cli.list_shares(*, cfg)`**Parameters** **cfg** (`lawwenda.Configuration`) –**Return type** str`lawwenda.lawwenda_cli.main()``lawwenda.lawwenda_cli.remove_share(*, cfg, name)`**Parameters**

- **cfg** (`lawwenda.Configuration`) –
- **name** (`str`) –

Return type None`lawwenda.lawwenda_cli.run_dev_server(*, cfg)`

Parameters `cfg` (`lawwenda.Configuration`) –

Return type `None`

1.1.9 lawwenda.server module

class `lawwenda.server.Server` (*, `cfgpath=None`)

Bases: `object`

Parameters `cfgpath` (`Optional[str]`) –

class `_ShareAppInst` (`share`)

Bases: `object`

Parameters `share` (`lawwenda.Share`) –

property `app`

Return type `lawwenda.fmapp.FmApp`

Return type `lawwenda.fmapp.FmApp`

Return type `lawwenda.fmapp.FmApp`

Return type `lawwenda.fmapp.FmApp`

property `is_active`

Return type `bool`

Return type `bool`

Return type `bool`

Return type `bool`

__get_app_for_share (`sharename`)

Parameters `sharename` (`str`) –

1.1.10 Module contents

User side API for reading and modifying Lawwenda configurations, creating shares, and more.

For most cases, instantiate `Configuration` and use some of its methods.

class `lawwenda.Configuration` (`cfgpath=None`)

Bases: `object`

Parameters `cfgpath` (`Optional[str]`) – The path of the configuration directory. Will be created on demand if it does not exist. Defaults to a location that is usual for your operating system.

__verify_valid_name ()

Parameters `name` (`str`) –

Return type `None`

add_share (`path`, *, `name`, `password`, `title=None`, `readonly=True`, `hide_by_patterns=()`, `hide_by_tags=()`, `include_by_patterns=None`, `include_by_tags=None`, `exclude_by_patterns=()`, `exclude_by_tags=()`, `exclude_hidden=False`, `active_until=None`)

Add a new share.

Parameters

- **path** (`str`) – The directory to share. See `Share.path`.
- **name** (`str`) – The unique name of the new share. See `Share.name`.

- **password** (*Optional[str]*) – The password to protect the share with.
- **title** (*Optional[str]*) – The share title. See *Share.title*.
- **readonly** (*bool*) – If to share in a read-only way. See *Share.readonly*.
- **hide_by_patterns** (*Iterable[str]*) – See *Share.hide_by_patterns*.
- **hide_by_tags** (*Iterable[str]*) – See *Share.hide_by_tags*.
- **include_by_patterns** (*Optional[Iterable[str]]*) – See *Share.include_by_patterns*.
- **include_by_tags** (*Optional[Iterable[str]]*) – See *Share.include_by_tags*.
- **exclude_by_patterns** (*Iterable[str]*) – See *Share.exclude_by_patterns*.
- **exclude_by_tags** (*Iterable[str]*) – See *Share.exclude_by_tags*.
- **exclude_hidden** (*bool*) – See *Share.exclude_hidden*.
- **active_until** (*Optional[datetime.datetime]*) – The optional expiration time of the share. See *Share.active_until*.

Return type *lawwenda.Share*

generate_wsgi ()

Return type *str*

get_share_by_name (*name*)

Return the share by a name (or *None* if it does not exist).

Parameters *name* (*str*) –

Return type *Optional[lawwenda.Share]*

get_shares ()

Return all shares that are currently part of this configuration.

Return type *List[lawwenda.Share]*

property path

The path of the configuration directory.

Return type *str*

Return type *str*

Return type *str*

Return type *str*

peek_share_cache_tag (*name*)

Parameters *name* (*str*) –

Return type *Optional[object]*

remove_share (*name*)

Parameters *name* (*str*) –

Return type *None*

run_dev_server ()

Return type `lawwenda.devserver._DevServerInfo`

```
class lawwenda.Share(path, *, configuration, name, title, readonly=True, hide_by_patterns=(),
                    hide_by_tags=(), include_by_patterns=None, include_by_tags=None, ex-
                    clude_by_patterns=(), exclude_by_tags=(), exclude_hidden=False, pass-
                    word_scrypt=None, password_salt=None, active_until_timestamp=None)
```

Bases: `object`

A directory path together with some parameters (e.g. for access control) for sharing via Lawwenda. Read the documentation for more about shares.

This class is not intended to be instantiated directly. You will get instances by some other api methods.

Parameters

- **path** (`str`) –
- **configuration** (`Configuration`) –
- **name** (`str`) –
- **title** (`str`) –
- **readonly** (`bool`) –
- **hide_by_patterns** (`Iterable[str]`) –
- **hide_by_tags** (`Iterable[str]`) –
- **include_by_patterns** (`Optional[Iterable[str]]`) –
- **include_by_tags** (`Optional[Iterable[str]]`) –
- **exclude_by_patterns** (`Iterable[str]`) –
- **exclude_by_tags** (`Iterable[str]`) –
- **exclude_hidden** (`bool`) –
- **password_scrypt** (`Optional[str]`) –
- **password_salt** (`Optional[str]`) –
- **active_until_timestamp** (`Optional[float]`) –

_set_cache_tag (`cachetag`)

Parameters **cachetag** (`object`) –

Return type `None`

_to_dict ()

property active_until

The expiration time of this share, or `None` for infinite.

Return type `t.Optional[datetime.datetime]`

Return type `t.Optional[datetime.datetime]`

Return type `t.Optional[datetime.datetime]`

Return type `t.Optional[datetime.datetime]`

property active_until_timestamp

Same as `active_until`, but as Unix timestamp.

Return type `t.Optional[float]`

Return type `t.Optional[float]`

Return type `t.Optional[float]`

Return type `t.Optional[float]`

property `cache_tag`

Return type `object`

Return type `object`

Return type `object`

Return type `object`

property `configuration`

The configuration that contains this share.

Return type *Configuration*

Return type *Configuration*

Return type *Configuration*

Return type *Configuration*

property `exclude_by_patterns`

A list of regular expressions of paths for excluding. A file or directory will be excluded if its path matches at least one of them. Those paths are always relative to *path*, always start with a `'`, but never end with a one (unless it is the root path).

Exclusions are enforced on backend side and not just a presentation aspect. There is no way for a client to work around that (unless there is a software bug).

Return type `t.Iterable[str]`

Return type `t.Iterable[str]`

Return type `t.Iterable[str]`

Return type `t.Iterable[str]`

property `exclude_by_tags`

A list of tags for excluding files and directories.

Exclusions are enforced on backend side and not just a presentation aspect. There is no way for a client to work around that (unless there is a software bug).

Return type `t.Iterable[str]`

Return type `t.Iterable[str]`

Return type `t.Iterable[str]`

Return type `t.Iterable[str]`

property `exclude_hidden`

If to consider 'hidden' flags of files or directories as exclusions.

Exclusions are enforced on backend side and not just a presentation aspect. There is no way for a client to work around that (unless there is a software bug).

Return type `bool`

Return type `bool`

Return type `bool`

Return type `bool`

property hide_by_patterns

A list of regular expressions of paths for hiding. A file or directory will be hidden if its path matches at least one of them. Those paths are always relative to *path*, always start with a '/', but never end with a one (unless it is the root path).

Note that hiding is not a security feature unless *exclude_hidden* is set.

Return type `t.Iterable[str]`

Return type `t.Iterable[str]`

Return type `t.Iterable[str]`

Return type `t.Iterable[str]`

property hide_by_tags

A list of tags for hiding files and directories. A file or directory will be hidden if it is tagged with at least one of them.

Note that hiding is not a security feature unless *exclude_hidden* is set.

Return type `t.Iterable[str]`

Return type `t.Iterable[str]`

Return type `t.Iterable[str]`

Return type `t.Iterable[str]`

property include_by_patterns

A list of regular expressions of paths for including explicitly. Those paths are always relative to *path*, always start with a '/', but never end with a one (unless it is the root path).

If this is specified, the share will switch from blacklist to whitelist. Everything that is not considered as included is implicitly considered as excluded.

Return type `t.Optional[t.Iterable[str]]`

Return type `t.Optional[t.Iterable[str]]`

Return type `t.Optional[t.Iterable[str]]`

Return type `t.Optional[t.Iterable[str]]`

property include_by_tags

A list of tags for including files and directories.

If this is specified, the share will switch from blacklist to whitelist. Everything that is not considered as included is implicitly considered as excluded.

Return type `t.Optional[t.Iterable[str]]`

Return type `t.Optional[t.Iterable[str]]`

Return type `t.Optional[t.Iterable[str]]`

Return type `t.Optional[t.Iterable[str]]`

property is_active

If this share is currently active (e.g. not yet expired; see *active_until*).

Return type `bool`

Return type `bool`

Return type `bool`

Return type `bool`

property name

The share name. This usually makes the last part of the url to this share. Is unique in the containing *configuration*.

Return type str

Return type str

Return type str

Return type str

property password_salt

The hash salt of the password for this share, if password protected. See also *password_script*.

Return type t.Optional[str]

Return type t.Optional[str]

Return type t.Optional[str]

Return type t.Optional[str]

property password_script

The script hash of the password for this share. Empty or *None* for disabled password protection. See also *password_salt*.

Return type t.Optional[str]

Return type t.Optional[str]

Return type t.Optional[str]

Return type t.Optional[str]

property path

The path of the share's root directory.

Return type str

Return type str

Return type str

Return type str

property readonly

If this share is restricted to only read actions (no removal, copying, uploading, editing, ... of the files and directories in *path*).

Return type bool

Return type bool

Return type bool

Return type bool

property title

The share title. This is an arbitrary text shown in the window title. Should not contain line breaks and should be short.

Return type str

Return type str

Return type str

Return type str

PYTHON MODULE INDEX

I

- lawwenda, [25](#)
- lawwenda.__aux, [3](#)
- lawwenda.__aux.projectinformation, [3](#)
- lawwenda.comm, [10](#)
- lawwenda.datafiles, [10](#)
- lawwenda.devserver, [10](#)
- lawwenda.fmapp, [11](#)
- lawwenda.fs, [13](#)
- lawwenda.lawwenda_cli, [23](#)
- lawwenda.previewer, [4](#)
- lawwenda.previewer.commonimages, [3](#)
- lawwenda.previewer.commonvideos, [4](#)
- lawwenda.search, [5](#)
- lawwenda.server, [25](#)
- lawwenda.sharehttpapp, [8](#)
- lawwenda.sharehttpapp.davprop, [7](#)

Symbols

_DevServerInfo (class in lawwenda.devserver), 10
 _DevServerThread (class in lawwenda.devserver), 10
 _PredicateFactory (class in lawwenda.fs), 23
 _RenderTemplateValue (class in lawwenda.fmapp), 12
 _TempZip__is_executable() (lawwenda.fmapp._TempZips._TempZip method), 12
 _TempZip__putnode() (lawwenda.fmapp._TempZips._TempZip method), 12
 _TempZips (class in lawwenda.fmapp), 12
 _TempZips._TempZip (class in lawwenda.fmapp), 12
 _TempZips__cleanup_loop() (lawwenda.fmapp._TempZips class method), 13
 __allowed_methods() (lawwenda.sharehttpapp.ShareHttpApp property), 8
 __auth() (lawwenda.fmapp.FmApp method), 11
 __decide() (lawwenda.search.ByGeoSearch method), 5
 __decide() (lawwenda.search.ByNameSearch method), 5
 __decide() (lawwenda.search.ByTagSearch method), 5
 __decide_unindexed() (lawwenda.search.DeeplySearch method), 5
 __decide_unindexed_by_content() (lawwenda.search.DeeplySearch method), 6
 __decide_unindexed_by_metadata() (lawwenda.search.DeeplySearch method), 6
 __dispatch_request() (lawwenda.fmapp.FmApp method), 11
 __dispatch_request_internals() (lawwenda.fmapp.FmApp method), 11
 __dispatch_request_normal() (lawwenda.fmapp.FmApp method), 11
 __ensure_authed() (lawwenda.fmapp.FmApp method), 11
 __ffmpeg_thumb() (lawwenda.preview.commonvideos.CommonVideosPreviewer method), 4
 __get_app_for_share() (lawwenda.server.Server method), 25
 __on_api_copymove() (lawwenda.fmapp.FmApp method), 11
 __path_to_fullpath() (lawwenda.fs.LocalFilesystem method), 22
 __query_indexed() (lawwenda.search.DeeplySearch method), 6
 __query_unindexed() (lawwenda.search.DeeplySearch method), 6
 __render_template() (lawwenda.fmapp.FmApp method), 11
 __render_template_text() (lawwenda.fmapp.FmApp method), 11
 __render_template_text_raw() (lawwenda.fmapp.FmApp method), 12
 __try_get_index() (lawwenda.search.DeeplySearch method), 6
 __verify_valid_name() (lawwenda.Configuration method), 25
 _abc_impl (lawwenda.preview.Previewer attribute), 4
 _abc_impl (lawwenda.preview.commonimages.CommonImagesPreviewer attribute), 3
 _abc_impl (lawwenda.preview.commonvideos.CommonVideosPreviewer attribute), 4
 _cleanup_thread (lawwenda.fmapp._TempZips attribute), 13
 _eval_predicate() (lawwenda.fs.AbstractFilesystemDecorator method), 13
 _filesystem() (lawwenda.fs.Filesystem.Node property), 14
 _fmt_cmd() (in module lawwenda.lawwenda_cli), 23

`_image_scale_to_thumbnail_size()`
(*lawwenda.previewer.Previewer* method), 4

`_lock` (*lawwenda.fmapp._TempZips* attribute), 13

`_method_copy()` (*lawwenda.sharehttpapp.ShareHttpApp* method), 8

`_method_delete()` (*lawwenda.sharehttpapp.ShareHttpApp* method), 8

`_method_get()` (*lawwenda.sharehttpapp.ShareHttpApp* method), 8

`_method_head()` (*lawwenda.sharehttpapp.ShareHttpApp* method), 9

`_method_mkcol()` (*lawwenda.sharehttpapp.ShareHttpApp* method), 9

`_method_move()` (*lawwenda.sharehttpapp.ShareHttpApp* method), 9

`_method_options()`
(*lawwenda.sharehttpapp.ShareHttpApp* method), 9

`_method_propfind()`
(*lawwenda.sharehttpapp.ShareHttpApp* method), 9

`_method_proppatch()`
(*lawwenda.sharehttpapp.ShareHttpApp* method), 9

`_method_put()` (*lawwenda.sharehttpapp.ShareHttpApp* method), 9

`_on_api_copy()` (*lawwenda.fmapp.FmApp* method), 12

`_on_api_delete()` (*lawwenda.fmapp.FmApp* method), 12

`_on_api_details()` (*lawwenda.fmapp.FmApp* method), 12

`_on_api_dir()` (*lawwenda.fmapp.FmApp* method), 12

`_on_api_known_tags()` (*lawwenda.fmapp.FmApp* method), 12

`_on_api_mkdir()` (*lawwenda.fmapp.FmApp* method), 12

`_on_api_move()` (*lawwenda.fmapp.FmApp* method), 12

`_on_api_rename()` (*lawwenda.fmapp.FmApp* method), 12

`_on_api_set_comment()`
(*lawwenda.fmapp.FmApp* method), 12

`_on_api_set_geo()` (*lawwenda.fmapp.FmApp* method), 12

`_on_api_set_rating()` (*lawwenda.fmapp.FmApp* method), 12

`_on_api_tag_entries()`
(*lawwenda.fmapp.FmApp* method), 12

`_on_api_thumbnail()` (*lawwenda.fmapp.FmApp* method), 12

`_on_api_untag_entries()`
(*lawwenda.fmapp.FmApp* method), 12

`_on_api_upload()` (*lawwenda.fmapp.FmApp* method), 12

`_on_api_zip()` (*lawwenda.fmapp.FmApp* method), 12

`_on_api_zip_download()`
(*lawwenda.fmapp.FmApp* method), 12

`_on_help()` (*lawwenda.fmapp.FmApp* method), 12

`_on_static()` (*lawwenda.fmapp.FmApp* method), 12

`_query_by_tree_traversal()` (in module *lawwenda.search*), 6

`_quickstart()` (in module *lawwenda.lawwenda_cli*), 23

`_set_cache_tag()` (*lawwenda.Share* method), 27

`_termstring_to_termlist()` (in module *lawwenda.search*), 6

`_to_dict()` (*lawwenda.Share* method), 27

`_zips` (*lawwenda.fmapp._TempZips* attribute), 13

A

`AbstractFilesystemDecorator` (class in *lawwenda.fs*), 13

`active_until()` (*lawwenda.Share* property), 27

`active_until_timestamp()` (*lawwenda.Share* property), 27

`add_share()` (in module *lawwenda.lawwenda_cli*), 23

`add_share()` (*lawwenda.Configuration* method), 25

`add_tag()` (*lawwenda.fs.Filesystem* method), 18

`add_tag()` (*lawwenda.fs.Filesystem.Node* method), 14

`add_tag()` (*lawwenda.fs.LocalFilesystem* method), 22

`app()` (*lawwenda.server.Server._ShareAppInst* property), 25

B

`basics_as_dict()` (*lawwenda.fs.Filesystem.Node* property), 14

`ByGeoSearch` (class in *lawwenda.search*), 5

`ByNameSearch` (class in *lawwenda.search*), 5

`ByTagSearch` (class in *lawwenda.search*), 5

`bytes()` (*lawwenda.fmapp._TempZips._TempZip* property), 13

C

`cache_tag()` (*lawwenda.Share* property), 28

`child_by_name()` (*lawwenda.fs.Filesystem.Node* method), 14

`child_nodes()` (*lawwenda.fs.ExcludeNodesFilesystemAdapter* method), 13

`child_nodes()` (*lawwenda.fs.Filesystem* method), 18

`child_nodes()` (*lawwenda.fs.Filesystem.Node* property), 14

`child_nodes()` (*lawwenda.fs.LocalFilesystem* method), 22

`CircularTraversalError`, 13

comment() (*lawwenda.fs.Filesystem method*), 18
 comment() (*lawwenda.fs.Filesystem.Node property*), 14
 comment() (*lawwenda.fs.LocalFilesystem method*), 22
 CommentDavProp (class in *lawwenda.sharehttpapp.davprop*), 7
 CommonImagesPreviewer (class in *lawwenda.preview.commonimages*), 3
 CommonVideosPreviewer (class in *lawwenda.preview.commonvideos*), 4
 Configuration (class in *lawwenda*), 25
 configuration() (*lawwenda.Share property*), 28
 console() (in module *lawwenda.lawwenda_cli*), 24
 copy_to() (*lawwenda.fs.Filesystem method*), 18
 copy_to() (*lawwenda.fs.Filesystem.Node method*), 14
 copy_to() (*lawwenda.fs.LocalFilesystem method*), 22
 create_filesystem() (in module *lawwenda.fs*), 23
 create_search() (in module *lawwenda.search*), 6
 create_tempzip() (*lawwenda.fmapp._TempZips class method*), 13

D

davname() (*lawwenda.sharehttpapp.davprop.DavProp property*), 7
 DavProp (class in *lawwenda.sharehttpapp.davprop*), 7
 DeeplySearch (class in *lawwenda.search*), 5
 delete() (*lawwenda.fs.Filesystem method*), 18
 delete() (*lawwenda.fs.Filesystem.Node method*), 14
 delete() (*lawwenda.fs.LocalFilesystem method*), 22
 describe_share() (in module *lawwenda.lawwenda_cli*), 24
 dirpath() (*lawwenda.fs.Filesystem.Node property*), 14

E

exclude_by_patterns() (*lawwenda.Share property*), 28
 exclude_by_tags() (*lawwenda.Share property*), 28
 exclude_hidden() (*lawwenda.Share property*), 28
 excludehidden() (*lawwenda.fs._PredicateFactory static method*), 23
 ExcludeNodesFilesystemAdapter (class in *lawwenda.fs*), 13
 exists() (*lawwenda.fs.Filesystem method*), 18
 exists() (*lawwenda.fs.Filesystem.Node property*), 14
 exists() (*lawwenda.fs.LocalFilesystem method*), 22

F

false() (*lawwenda.fs._PredicateFactory static method*), 23
 Filesystem (class in *lawwenda.fs*), 13
 Filesystem.Node (class in *lawwenda.fs*), 14
 Filesystem.ReadHandle (class in *lawwenda.fs*), 17

Filesystem.WriteHandle (class in *lawwenda.fs*), 18
 find_data_file() (in module *lawwenda.datafiles*), 10
 FmApp (class in *lawwenda.fmapp*), 11
 full_as_dict() (*lawwenda.fs.Filesystem.Node property*), 14

G

generate_wsgi() (in module *lawwenda.lawwenda_cli*), 24
 generate_wsgi() (*lawwenda.Configuration method*), 26
 geo() (*lawwenda.fs.Filesystem method*), 18
 geo() (*lawwenda.fs.Filesystem.Node property*), 14
 geo() (*lawwenda.fs.LocalFilesystem method*), 22
 geo_obj() (*lawwenda.fs.Filesystem.Node property*), 15
 GeoDavProp (class in *lawwenda.sharehttpapp.davprop*), 7
 get_for_node() (*lawwenda.sharehttpapp.davprop.CommentDavProp method*), 7
 get_for_node() (*lawwenda.sharehttpapp.davprop.DavProp method*), 7
 get_for_node() (*lawwenda.sharehttpapp.davprop.GeoDavProp method*), 7
 get_for_node() (*lawwenda.sharehttpapp.davprop.MtimeDavProp method*), 7
 get_for_node() (*lawwenda.sharehttpapp.davprop.RatingDavProp method*), 7
 get_for_node() (*lawwenda.sharehttpapp.davprop.ResourceTypeDavProp method*), 7
 get_for_node() (*lawwenda.sharehttpapp.davprop.SizeDavProp method*), 8
 get_for_node() (*lawwenda.sharehttpapp.davprop.TagsDavProp method*), 8
 get_prop_by_davname() (in module *lawwenda.sharehttpapp.davprop*), 8
 get_readhandle() (*lawwenda.fs.ExcludeNodesFilesystemAdapter method*), 13
 get_readhandle() (*lawwenda.fs.Filesystem method*), 18
 get_readhandle() (*lawwenda.fs.LocalFilesystem method*), 22
 get_running_dev_servers() (in module *lawwenda.devserver*), 11
 get_share_by_name() (*lawwenda.Configuration method*), 26
 get_shares() (*lawwenda.Configuration method*), 26
 get_tempzip() (*lawwenda.fmapp._TempZips class method*), 13
 get_writehandle() (*lawwenda.fs.ExcludeNodesFilesystemAdapter method*), 13

`get_writehandle()` (*lawwenda.fs.Filesystem method*), 18
`get_writehandle()` (*lawwenda.fs.LocalFilesystem method*), 22
`get_writehandle()` (*lawwenda.fs.ReadOnlyFilesystemAdapter method*), 23

H

`has_preview()` (*lawwenda.fs.Filesystem method*), 19
`has_preview()` (*lawwenda.fs.Filesystem.Node property*), 15
`has_preview()` (*lawwenda.fs.LocalFilesystem method*), 22
`has_thumbnail()` (*lawwenda.fs.Filesystem method*), 19
`has_thumbnail()` (*lawwenda.fs.Filesystem.Node property*), 15
`has_thumbnail()` (*lawwenda.fs.LocalFilesystem method*), 22
`hide_by_patterns()` (*lawwenda.Share property*), 28
`hide_by_tags()` (*lawwenda.Share property*), 29
`HideNodesFilesystemAdapter` (class in *lawwenda.fs*), 21

I

`icon_name()` (*lawwenda.fs.Filesystem.Node property*), 15
`include_by_patterns()` (*lawwenda.Share property*), 29
`include_by_tags()` (*lawwenda.Share property*), 29
`is_active()` (*lawwenda.server.Server._ShareAppInst property*), 25
`is_active()` (*lawwenda.Share property*), 29
`is_dir()` (*lawwenda.fs.Filesystem method*), 19
`is_dir()` (*lawwenda.fs.Filesystem.Node property*), 15
`is_dir()` (*lawwenda.fs.LocalFilesystem method*), 22
`is_file()` (*lawwenda.fs.Filesystem method*), 19
`is_file()` (*lawwenda.fs.Filesystem.Node property*), 15
`is_file()` (*lawwenda.fs.LocalFilesystem method*), 22
`is_hidden()` (*lawwenda.fs.Filesystem method*), 19
`is_hidden()` (*lawwenda.fs.Filesystem.Node property*), 15
`is_hidden()` (*lawwenda.fs.HideNodesFilesystemAdapter method*), 21
`is_hidden()` (*lawwenda.fs.LocalFilesystem method*), 22
`is_link()` (*lawwenda.fs.Filesystem method*), 19
`is_link()` (*lawwenda.fs.Filesystem.Node property*), 15
`is_link()` (*lawwenda.fs.LocalFilesystem method*), 22

`is_previewable()` (*lawwenda.preview.commonimages.CommonImagesPreviewer method*), 3
`is_previewable()` (*lawwenda.preview.commonvideos.CommonVideosPreviewer method*), 4
`is_previewable()` (*lawwenda.preview.Previewer method*), 4
`is_thumbnailable()` (*lawwenda.preview.commonimages.CommonImagesPreviewer method*), 3
`is_thumbnailable()` (*lawwenda.preview.commonvideos.CommonVideosPreviewer method*), 4
`is_thumbnailable()` (*lawwenda.preview.Previewer method*), 4
`is_writable()` (*lawwenda.fs.Filesystem.Node property*), 15
`is_writable()` (*lawwenda.sharehttpapp.davprop.DavProp property*), 7
`IsAbleRequest` (class in *lawwenda.preview*), 4

J

`JsonedResponse` (class in *lawwenda.comm*), 10

K

`known_tags()` (*lawwenda.fs.Filesystem method*), 19
`known_tags()` (*lawwenda.fs.LocalFilesystem method*), 22

L

`lawwenda` module, 25
`lawwenda._aux` module, 3
`lawwenda._aux.projectinformation` module, 3
`lawwenda.comm` module, 10
`lawwenda.datafiles` module, 10
`lawwenda.devserver` module, 10
`lawwenda.fmapp` module, 11
`lawwenda.fs` module, 13
`lawwenda.lawwenda_cli` module, 23
`lawwenda.preview` module, 4
`lawwenda.preview.commonimages` module, 3
`lawwenda.preview.commonvideos` module, 4

lawwenda.search
 module, 5
 lawwenda.server
 module, 25
 lawwenda.sharehttpapp
 module, 8
 lawwenda.sharehttpapp.davprop
 module, 7
 list_shares() (in module lawwenda.lawwenda_cli), 24
 LocalFilesystem (class in lawwenda.fs), 21

M

main() (in module lawwenda.lawwenda_cli), 24
 mimetype (lawwenda.previewer.IsAbleRequest attribute), 4
 mimetype() (lawwenda.fs.Filesystem method), 19
 mimetype() (lawwenda.fs.Filesystem.Node property), 15
 mimetype() (lawwenda.fs.LocalFilesystem method), 22
 mkdir() (lawwenda.fs.Filesystem method), 19
 mkdir() (lawwenda.fs.Filesystem.Node method), 16
 mkdir() (lawwenda.fs.LocalFilesystem method), 22
 module
 lawwenda, 25
 lawwenda._aux, 3
 lawwenda._aux.projectinformation, 3
 lawwenda.comm, 10
 lawwenda.datafiles, 10
 lawwenda.devserver, 10
 lawwenda.fmapp, 11
 lawwenda.fs, 13
 lawwenda.lawwenda_cli, 23
 lawwenda.previewer, 4
 lawwenda.previewer.commonimages, 3
 lawwenda.previewer.commonvideos, 4
 lawwenda.search, 5
 lawwenda.server, 25
 lawwenda.sharehttpapp, 8
 lawwenda.sharehttpapp.davprop, 7
 move_to() (lawwenda.fs.Filesystem method), 19
 move_to() (lawwenda.fs.Filesystem.Node method), 16
 move_to() (lawwenda.fs.LocalFilesystem method), 22
 mtime() (lawwenda.fs.Filesystem method), 19
 mtime() (lawwenda.fs.Filesystem.Node property), 16
 mtime() (lawwenda.fs.LocalFilesystem method), 22
 mtime_ts() (lawwenda.fs.Filesystem.Node property), 16
 MtimeDavProp (class in lawwenda.sharehttpapp.davprop), 7

N

name (lawwenda.previewer.IsAbleRequest attribute), 4

name() (lawwenda.fs.Filesystem.Node property), 16
 name() (lawwenda.Share property), 29
 node_by_path() (lawwenda.fs.Filesystem method), 19

P

p_descending() (lawwenda.fs._PredicateFactory static method), 23
 p_not() (lawwenda.fs._PredicateFactory static method), 23
 p_or() (lawwenda.fs._PredicateFactory static method), 23
 p_regexp() (lawwenda.fs._PredicateFactory static method), 23
 p_tag() (lawwenda.fs._PredicateFactory static method), 23
 parent_node() (lawwenda.fs.Filesystem.Node property), 16
 password_salt() (lawwenda.Share property), 30
 password_scrypt() (lawwenda.Share property), 30
 path() (lawwenda.Configuration property), 26
 path() (lawwenda.fs.Filesystem.Node property), 16
 path() (lawwenda.Share property), 30
 peek_share_cache_tag() (lawwenda.Configuration method), 26
 preview_html() (lawwenda.fs.Filesystem method), 20
 preview_html() (lawwenda.fs.Filesystem.Node property), 16
 preview_html() (lawwenda.fs.LocalFilesystem method), 22
 preview_html() (lawwenda.previewer.commonimages.CommonImages method), 3
 preview_html() (lawwenda.previewer.commonvideos.CommonVideosP method), 4
 preview_html() (lawwenda.previewer.Previewer method), 4
 Previewer (class in lawwenda.previewer), 4

Q

query() (lawwenda.search.ByGeoSearch method), 5
 query() (lawwenda.search.ByNameSearch method), 5
 query() (lawwenda.search.ByTagSearch method), 5
 query() (lawwenda.search.DeeplySearch method), 6
 query() (lawwenda.search.Search method), 6

R

rating() (lawwenda.fs.Filesystem method), 20
 rating() (lawwenda.fs.Filesystem.Node property), 16
 rating() (lawwenda.fs.LocalFilesystem method), 22
 RatingDavProp (class in lawwenda.sharehttpapp.davprop), 7
 read_file() (lawwenda.fs.Filesystem method), 20

`read_file()` (*lawwenda.fs.Filesystem.Node* method), 16
`read_file()` (*lawwenda.fs.LocalFilesystem* method), 22
`readonly()` (*lawwenda.Share* property), 30
`ReadOnlyFilesystemAdapter` (class in *lawwenda.fs*), 22
`register_prop()` (in module *lawwenda.sharehttpapp.davprop*), 8
`remove_share()` (in module *lawwenda.lawwenda_cli*), 24
`remove_share()` (*lawwenda.Configuration* method), 26
`remove_tag()` (*lawwenda.fs.Filesystem* method), 20
`remove_tag()` (*lawwenda.fs.Filesystem.Node* method), 16
`remove_tag()` (*lawwenda.fs.LocalFilesystem* method), 22
`Request` (class in *lawwenda.comm*), 10
`ResourceTypeDavProp` (class in *lawwenda.sharehttpapp.davprop*), 7
`rootnode()` (*lawwenda.fs.Filesystem* property), 20
`run()` (*lawwenda.devserver._DevServerThread* method), 11
`run_dev_server()` (in module *lawwenda.devserver*), 11
`run_dev_server()` (in module *lawwenda.lawwenda_cli*), 24
`run_dev_server()` (*lawwenda.Configuration* method), 26

S

`sanitize_abspath()` (*lawwenda.fs.Filesystem* static method), 20
`Search` (class in *lawwenda.search*), 6
`Server` (class in *lawwenda.server*), 25
`Server._ShareAppInst` (class in *lawwenda.server*), 25
`set_comment()` (*lawwenda.fs.Filesystem* method), 20
`set_comment()` (*lawwenda.fs.Filesystem.Node* method), 17
`set_comment()` (*lawwenda.fs.LocalFilesystem* method), 22
`set_geo()` (*lawwenda.fs.Filesystem* method), 20
`set_geo()` (*lawwenda.fs.Filesystem.Node* method), 17
`set_geo()` (*lawwenda.fs.LocalFilesystem* method), 22
`set_rating()` (*lawwenda.fs.Filesystem* method), 21
`set_rating()` (*lawwenda.fs.Filesystem.Node* method), 17
`set_rating()` (*lawwenda.fs.LocalFilesystem* method), 22
`Share` (class in *lawwenda*), 27
`ShareHttpApp` (class in *lawwenda.sharehttpapp*), 8

`shutdown()` (*lawwenda.devserver._DevServerInfo* method), 10
`size()` (*lawwenda.fs.Filesystem* method), 21
`size()` (*lawwenda.fs.Filesystem.Node* property), 17
`size()` (*lawwenda.fs.LocalFilesystem* method), 22
`SizeDavProp` (class in *lawwenda.sharehttpapp.davprop*), 8

T

`tags()` (*lawwenda.fs.Filesystem* method), 21
`tags()` (*lawwenda.fs.Filesystem.Node* property), 17
`tags()` (*lawwenda.fs.LocalFilesystem* method), 22
`TagsDavProp` (class in *lawwenda.sharehttpapp.davprop*), 8
`tagstring()` (*lawwenda.fs.Filesystem.Node* property), 17
`thumbheight` (*lawwenda.preview.Previewer* attribute), 5
`thumbnail()` (*lawwenda.fs.Filesystem* method), 21
`thumbnail()` (*lawwenda.fs.Filesystem.Node* method), 17
`thumbnail()` (*lawwenda.fs.LocalFilesystem* method), 22
`thumbnail()` (*lawwenda.preview.commonimages.CommonImagesPreviewer* method), 3
`thumbnail()` (*lawwenda.preview.commonvideos.CommonVideosPreviewer* method), 4
`thumbnail()` (*lawwenda.preview.Previewer* method), 5
`thumbsize` (*lawwenda.preview.Previewer* attribute), 5
`thumbwidth` (*lawwenda.preview.Previewer* attribute), 5
`title()` (*lawwenda.Share* property), 30
`traverse_dir()` (*lawwenda.fs.Filesystem.Node* method), 17
`try_get_fullpath()` (*lawwenda.fs.Filesystem* method), 21
`try_get_fullpath()` (*lawwenda.fs.Filesystem.Node* method), 17
`try_get_fullpath()` (*lawwenda.fs.LocalFilesystem* method), 22

U

`unescape()` (*lawwenda.fmapp._RenderTemplateValue* property), 12
`url()` (*lawwenda.devserver._DevServerInfo* property), 10
`url_internals_name` (*lawwenda.fmapp.FmApp* attribute), 12

W

`wait_stopped()` (*lawwenda.devserver._DevServerInfo*

method), 10
write_file() (*lawwenda.fs.Filesystem method*), 21
write_file() (*lawwenda.fs.Filesystem.Node*
method), 17
write_file() (*lawwenda.fs.LocalFilesystem*
method), 22