

Contents

1	NAME	2
2	SYNOPSIS	2
3	DESCRIPTION	2
4	PUBLIC METHODS	2

1 NAME

OpenSLX::ScopedResource - provides a helper class that implements the 'resource-acquisition-by-definition' pattern.

2 SYNOPSIS

```
{ # some scope

    my $distroSession = OpenSLX::ScopedResource->new({
        name      => 'distro::session',
        acquire => sub { $distro->startSession(); 1 },
        release => sub { $distro->finishSession(); 1 },
    });

    die $@ if ! eval {
        # do something dangerous and unpredictable here:
        doRandomStuff();
        1;
    };

} # the distro-session will be cleanly finished, no matter if we died or not
```

3 DESCRIPTION

The class `ScopedResource` wraps any resource such that the resource will be acquired when an object of this class is created. Whenever the `ScopedResource` object is being destroyed (e.g. by leaving scope) the wrapped resource will automatically be released.

The main purpose of this class is to make it simple to implement reliable resource acquisition and release management even if the structure of the code that refers to that resource is rather complex.

Furthermore, this class handles cases where the script handling those resources is spread across different process and/or makes us of signal handlers.

4 PUBLIC METHODS

new(\$params)

Creates a `ScopedResource` object for the resource specified by the given *\$params*.

As part of creation of the object, the resource will be acquired.

The *\$params*-hashref requires the following entries:

name

Gives a name for the wrapped resource. This is just used in log messages concerning the acquisition and release of that resource.

acuire

Gives the code that is going to be executed in order to acquire the resource.

release

Gives the code that is going to be executed in order to release the resource.

DESTROY()

Releases the resource (if it had been acquired by this process) and cleans up.