

---

# **PyXenon Documentation**

***Release 2.2.0***

**Johan Hidding, Stefan Verhoeven**

**Sep 13, 2019**



---

## Contents

---

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>Quick Start</b>                                | <b>1</b>  |
| 1.1      | Starting the server . . . . .                     | 1         |
| 1.2      | Writing to a remote filesystem . . . . .          | 1         |
| 1.3      | Running a script . . . . .                        | 2         |
| 1.4      | Retrieving the result . . . . .                   | 2         |
| <b>2</b> | <b>Advanced: Streaming &amp; Interactive jobs</b> | <b>3</b>  |
| 2.1      | Example: an online job . . . . .                  | 3         |
| 2.2      | Protocol definitions . . . . .                    | 5         |
| <b>3</b> | <b>Adaptors</b>                                   | <b>7</b>  |
| 3.1      | File System . . . . .                             | 7         |
| 3.2      | Scheduler . . . . .                               | 10        |
| <b>4</b> | <b>API</b>  | <b>17</b> |
| 4.1      | The Server . . . . .                              | 17        |
| 4.2      | File Systems . . . . .                            | 18        |
| 4.3      | Schedulers . . . . .                              | 22        |
| 4.4      | Credentials . . . . .                             | 25        |
| 4.5      | Exceptions . . . . .                              | 26        |
| <b>5</b> | <b>Installing</b>                                 | <b>29</b> |
| <b>6</b> | <b>Indices and tables</b>                         | <b>31</b> |
|          | <b>Python Module Index</b>                        | <b>33</b> |
|          | <b>Index</b>                                      | <b>35</b> |



# CHAPTER 1

---

## Quick Start

---

We like to test Xenon against a [Docker](#) image: [nlesc/xenon-slurm](#). If you have docker all setup, you can run this image as follows:

```
$ docker pull nlesc/xenon-slurm
...
$ docker run --detach --publish 10022:22 nlesc/xenon-slurm
```

Try logging onto this image by *ssh*, to make sure everything works. The username is *xenon*, the password *javagat*:

```
$ ssh localhost -p 10022 -l xenon
xenon@localhost's password: <javagat>
$ exit
Connection to localhost closed.
```

## 1.1 Starting the server

To get anything done in PyXenon, we need to start the GRPC server:

```
import xenon

xenon.init()
```

## 1.2 Writing to a remote filesystem

Next, let's try to copy a file to the container. We need credentials to access anything on the remote side.

```
from xenon import PasswordCredential, FileSystem

credential = PasswordCredential(
```

(continues on next page)

(continued from previous page)

```
username='xenon',
password='javagat')

remotefs = FileSystem.create(
    'sftp', location='localhost:10022',
    password_credential=credential)
```

We can write to a file by streaming. The second argument to `write_to_file()` should be an iterable. It will be read in a separate thread, so it is allowed to be blocking. Here we'll do nothing so fancy:

```
from xenon import Path

target = Path('hello.sh')

if remotefs.exists(target):
    remotefs.delete(target)

remotefs.write_to_file(
    target,
    [b'#!/bin/sh\n',
     b'echo "Hello, World!"\n'])
```

## 1.3 Running a script

The remote machine runs a SLURM job scheduler. We describe a job in a `JobDescription` object. This seems a bit long-winded, but in practice you'll be reusing the descriptions a lot.

```
from xenon import Scheduler, JobDescription

scheduler = Scheduler.create(
    adaptor='slurm',
    location='ssh://localhost:10022',
    password_credential=credential)

job_description = JobDescription(
    executable='/bin/sh',
    arguments=['hello.sh'],
    stdout='result.txt')

job = scheduler.submit_batch_job(job_description)

state = scheduler.wait_until_done(job)
print(state)
```

## 1.4 Retrieving the result

Just as we can write data by sending an iterable, we can read data from a file and receive a generator yielding bytes objects. Here we realize the transfer by joining the data chunks into a string:

```
text = ''.join(chunk.decode() for chunk in
    remotefs.read_from_file(Path('result.txt')))
print(text)
```

---

## Advanced: Streaming & Interactive jobs

---

In several cases it is desirable to stream data from/to interactive jobs as well as data to a remote filesystem. The GRPC API has built-in support for asynchronous streaming through many simultaneous requests. In Python this API is exposed in terms of generators.

### 2.1 Example: an online job

In this example we'll show how to obtain bi-directional communication with an online job. An online job is started with `Scheduler.submit_online_job()`.

#### 2.1.1 Streaming input, a.k.a. The Halting Problem

We need to stream input to the online job. In the *Quick Start*, we saw that we could send data to a stream by simply giving a list of bytes objects. Here we aim a bit more advanced to play a kind of real-time ping-pong with a remote process. We need to provide *PyXenon* with an generator that pulls its messages from a queue. The GRPC module ensures that this generator is being run asynchronously from the main thread.

The tricky part is that we need to be able to tell the generator when the work is done and no more input is to be expected. We could have it receive strings and make it check for end-of-file messages in some way, but in essence we'll always have to define a little protocol to deal with the finiteness of the generator's life. To make this explicit we define a little 2-tuple micro-language:

| message                  | action               |
|--------------------------|----------------------|
| ('msg', <value: string>) | yield value.encode() |
| ('end', None)            | return               |

Implementing this:

```
from queue import Queue

def make_input_stream():
    input_queue = Queue()

    def input_stream():
        while True:
            cmd, value = input_queue.get()
            if cmd == 'end':
                input_queue.task_done()
                return
            elif cmd == 'msg':
                yield value.encode()
                input_queue.task_done()

    return input_queue, input_stream
```

### 2.1.2 Reading output

The return-value of `submit_online_job()` is an iterator yielding objects of type *SubmitOnlineJobResponse*. These objects have a `stdout` field containing (binary) data that the job wrote to standard output, as well as a `stderr` field containing data written to standard error. For any message either field may be empty or not. In this example we're only interested in data from `stdout`:

```
def get_stdout(stream):
    return stream.next().stdout.decode()
```

### 2.1.3 The “remote” script

For the purpose of this example, we have defined a small Python `rot13` program:

Listing 1: `rot13.py`

```
import codecs

try:
    while True:
        line = input()
        print(codecs.encode(line, 'rot_13'))

except EOFError:
    pass
```

### 2.1.4 Defining the job

Online job descriptions are the same as normal job descriptions.

```
# our input lines
input_lines = [
    "Zlfgvp aboyr tnf,",
    "Urnil lrg syrrgvat sebz tenfc,",
    "Oyhr yvyr oheavat vpr."
```

(continues on next page)



(continued from previous page)

```

]

# the job description, make sure you run the script from the examples
# directory!
job_description = xenon.JobDescription(
    executable='python',
    arguments=['rot13.py'],
    queue_name='multi')

```

## 2.1.5 Putting it together

The rest is history.

```

import xenon

# start the xenon-grpc server
xenon.init()

# on the local adaptor
with xenon.Scheduler.create(adaptor='local') as scheduler:
    input_queue, input_stream = make_input_stream()

    # submit an interactive job, this gets us the job-id and a stream
    # yielding job output from stdout and stderr.
    job, output_stream = scheduler.submit_interactive_job(
        description=job_description, stdin_stream=input_stream())

    # next we feed the input_queue with messages
    try:
        for line in input_lines:
            print("[sending]   " + line)
            input_queue.put(('msg', line + '\n'))
            msg = get_stdout(output_stream)
            print("[received]  " + msg)

    # make sure to close our end whatever may happen
    finally:
        input_queue.put(('end', None))
        input_queue.join()

    scheduler.wait_until_done(job)

```

## 2.2 Protocol definitions

It can be instructive to see what the GRPC protocol with respect to interactive jobs looks like.

```

message SubmitInteractiveJobRequest {
    Scheduler scheduler = 1;
    JobDescription description = 2;
    bytes stdin = 3;
}

```

(continues on next page)

(continued from previous page)

```
message SubmitInteractiveJobResponse {
    Job job = 1;
    bytes stdout = 2;
    bytes stderr = 3;
}

service SchedulerService {
    rpc submitInteractiveJob(
        stream SubmitInteractiveJobRequest)
        returns (stream SubmitInteractiveJobResponse) {}
}
```

In *PyXenon* the remote procedure call `submitInteractiveJob` is wrapped to the method `submit_interactive_job()` of the `Scheduler` class. Note that the `SubmitInteractiveJobRequest` specifies (next to the scheduler, which is obtained from `self` in the method call) the job description and bytes for standard input. Requests of this type are streamed. This means that GRPC expects to get an iterator of `SubmitInteractiveJobRequest` objects.

The *PyXenon* `submit_interactive_job()` method separates the job-description and input-stream arguments. Sending the scheduler and description fields in the first request, followed up by a sequence of requests where only the `stdin` field is specified. This latter sequence is yielded from the `stdin_stream` argument.

Similarly, the first item in the output stream is guaranteed to only contain the job-id, this first item is available immediately. Subsequent calls to `next(output_stream)` will block until output is available. The `submit_interactive_job()` method takes the first item of the iterator, and extracts the job-id. The user receives a tuple with the extracted job-id and the iterator.

This section contains the adaptor documentation which is generated from the information provided by the adaptors themselves.

### Contents

- *Adaptors*
  - *File System*
    - \* *S3*
    - \* *File*
    - \* *Sftp*
    - \* *Ftp*
    - \* *Webdav*
  - *Scheduler*
    - \* *Local*
    - \* *Ssh*
    - \* *At*
    - \* *Slurm*
    - \* *Gridengine*
    - \* *Torque*

## 3.1 File System

---

**Note:** Supported property names should be prefixed with

---

"xenon.adaptors.filesystems". We've left this prefix out to improve readability of the tables.

### 3.1.1 S3

The S3 adaptor uses Apache JClouds to talk to s3 and others. To authenticate use PasswordCredential with access key id as username and secret access key as password

| field                              | value                     |
|------------------------------------|---------------------------|
| supports_third_party_copy          | False                     |
| can_create_symboliclinks           | False                     |
| can_read_symboliclinks             | False                     |
| is_connectionless                  | True                      |
| supported_credentials              | <i>PasswordCredential</i> |
| can_append                         | False                     |
| supports_reading_posix_permissions | False                     |
| supports_setting_posix_permissions | False                     |
| supports_rename                    | False                     |
| needs_size_beforehand              | True                      |

**location string:**

- `http[s]://host[:port]/bucketname[/workdir]`
- `https://s3.region.amazonaws.com/bucketname[/workdir]`

supported properties:

| name          | description   | data_type | default    |
|---------------|---|-----------|------------|
| s3.bufferSize | The buffer size to use when copying files (in bytes). | size      | <i>64K</i> |

### 3.1.2 File

This is the local file adaptor that implements file functionality for local access.

| field                              | value                    |
|------------------------------------|--------------------------|
| supports_third_party_copy          | False                    |
| can_create_symboliclinks           | True                     |
| can_read_symboliclinks             | True                     |
| is_connectionless                  | True                     |
| supported_credentials              | <i>DefaultCredential</i> |
| can_append                         | True                     |
| supports_reading_posix_permissions | True                     |
| supports_setting_posix_permissions | True                     |
| supports_rename                    | True                     |
| needs_size_beforehand              | False                    |

**location string:**

- *(null)*

- (empty string)
- [/workdir]
- driveletter:[/workdir]

supported properties:

| name            | description   | data_type | default |
|-----------------|---|-----------|---------|
| file.bufferSize | The buffer size to use when copying files (in bytes). | size      | 64K     |

### 3.1.3 Sftp

The SFTP adaptor implements all file access functionality to remote SFTP servers

| field                               | value  |
|-------------------------------------|--|
| supports_third_party_copy           | False  |
| can_create_symboliclinks            | True   |
| can_read_symboliclinks              | True   |
| is_connectionless                   | False  |
| supported_credentials               | <i>DefaultCredential, CertificateCredential, PasswordCredential, CredentialMap</i> |
| can_append                          | True   |
| sup-ports_reading_posix_permissions | True   |
| sup-ports_setting_posix_permissions | True   |
| supports_rename                     | True   |
| needs_size_beforehand               | False  |

**location string:**

- *host[:port][/workdir]*

supported properties:

| name                       | description  | data_type | de-fault     |
|----------------------------|--|-----------|--------------|
| sftp.strictHostKeyChecking | Enable strict host key checking.   | boolean   | <i>true</i>  |
| sftp.loadKnownHosts        | Load the standard known_hosts file.  | boolean   | <i>true</i>  |
| sftp.loadSshConfig         | Load the OpenSSH config file.  | boolean   | <i>true</i>  |
| sftp.agent                 | Use a (local) ssh-agent.   | boolean   | <i>false</i> |
| sftp.agentForwarding       | Use ssh-agent forwarding when setting up a connection.                     | boolean   | <i>false</i> |
| sftp.connection.timeout    | The timeout for creating and authenticating connections (in milliseconds). | natural   | <i>10000</i> |
| sftp.bufferSize            | The buffer size to use when copying files (in bytes).                      | size      | <i>64K</i>   |

### 3.1.4 Ftp

The FTP adaptor implements file access on remote ftp servers.

| field                              | value  |
|------------------------------------|--|
| supports_third_party_copy          | False  |
| can_create_symboliclinks           | False  |
| can_read_symboliclinks             | True   |
| is_connectionless                  | False  |
| supported_credentials              | <i>DefaultCredential, PasswordCredential</i> |
| can_append                         | True   |
| supports_reading_posix_permissions | True   |
| supports_setting_posix_permissions | False  |
| supports_rename                    | True   |
| needs_size_beforehand              | False  |

**location string:**

- *host[:port][/workdir]*

supported properties:

| name           | description   | data_type | default    |
|----------------|---|-----------|------------|
| ftp.bufferSize | The buffer size to use when copying files (in bytes). | size      | <i>64K</i> |

### 3.1.5 Webdav

The webdav file adaptor implements file access to remote webdav servers.

| field                              | value  |
|------------------------------------|--|
| supports_third_party_copy          | False  |
| can_create_symboliclinks           | False  |
| can_read_symboliclinks             | False  |
| is_connectionless                  | True   |
| supported_credentials              | <i>DefaultCredential, PasswordCredential</i> |
| can_append                         | False  |
| supports_reading_posix_permissions | False  |
| supports_setting_posix_permissions | False  |
| supports_rename                    | True   |
| needs_size_beforehand              | False  |

**location string:**

- *http://host[:port][/workdir]*
- *https://host[:port][/workdir]*

supported properties:

| name              | description   | data_type | default    |
|-------------------|---|-----------|------------|
| webdav.bufferSize | The buffer size to use when copying files (in bytes). | size      | <i>64K</i> |

## 3.2 Scheduler

---

**Note:** Supported property names should be prefixed with

---

"xenon.adaptors.schedulers". We've left this prefix out to improve readability of the tables.

### 3.2.1 Local

The local jobs adaptor implements all functionality by emulating a local queue.

| field                 | value                    |
|-----------------------|--------------------------|
| is_embedded           | True                     |
| supports_interactive  | True                     |
| supports_batch        | True                     |
| uses_file_system      | True                     |
| supported_credentials | <i>DefaultCredential</i> |

**location string:**

- *[/workdir]*

supported properties:

| name                                | description  | data_type | de-<br>fault |
|-------------------------------------|--|-----------|--------------|
| local.queue.pollingDelay            | The polling delay for monitoring running jobs (in milliseconds). | long      | <i>1000</i>  |
| local.queue.multi.maxConcurrentJobs | The maximum number of concurrent jobs in the multiq.             | integer   | <i>4</i>     |

### 3.2.2 Ssh

The SSH job adaptor implements all functionality to start jobs on ssh servers.

| field                 | value  |
|-----------------------|--|
| is_embedded           | True   |
| supports_interactive  | True   |
| supports_batch        | True   |
| uses_file_system      | True   |
| supported_credentials | <i>DefaultCredential, CertificateCredential, PasswordCredential, CredentialMap</i> |

**location string:**

- *host[:port][[/workdir][ via:otherhost[:port]]\*]*

supported properties:

| name                              | description  | data_type | de-<br>fault |
|-----------------------------------|--|-----------|--------------|
| ssh.strictHostKeyChecking         | Enable strict host key checking.   | boolean   | <i>true</i>  |
| ssh.loadKnownHosts                | Load the standard known_hosts file.  | boolean   | <i>true</i>  |
| ssh.loadSshConfig                 | Load the OpenSSH config file.  | boolean   | <i>true</i>  |
| ssh.agent                         | Use a (local) ssh-agent.   | boolean   | <i>false</i> |
| ssh.agentForwarding               | Use ssh-agent forwarding   | boolean   | <i>false</i> |
| ssh.timeout                       | The timeout for the connection setup and authentication (in milliseconds). | long      | <i>10000</i> |
| ssh.queue.pollingDelay            | The polling delay for monitoring running jobs (in milliseconds).           | long      | <i>1000</i>  |
| ssh.queue.multi.maxConcurrentJobs | The maximum number of concurrent jobs in the multiq..                      | integer   | <i>4</i>     |

### 3.2.3 At

The At Adaptor submits jobs to an at scheduler. This adaptor uses either the local or the ssh scheduler adaptor to run commands on the machine running at, and the file or the stfp filesystem adaptor to gain access to the filesystem of that machine.

| field                 | value  |
|-----------------------|--|
| is_embedded           | False  |
| supports_interactive  | False  |
| supports_batch        | True   |
| uses_file_system      | True   |
| supported_credentials | <i>DefaultCredential, CertificateCredential, PasswordCredential, CredentialMap</i> |

#### location string:

- *local://[/workdir]*
- *ssh://host[:port][[/workdir]][ via:otherhost[:port]]\**

supported properties:

| name                                | description  | data_type | de-<br>fault |
|-------------------------------------|--|-----------|--------------|
| at.poll.delay                       | Number of milliseconds between polling the status of a job.                | long      | <i>1000</i>  |
| ssh.strictHostKeyChecking           | Enable strict host key checking.   | boolean   | <i>true</i>  |
| ssh.loadKnownHosts                  | Load the standard known_hosts file.  | boolean   | <i>true</i>  |
| ssh.loadSshConfig                   | Load the OpenSSH config file.  | boolean   | <i>true</i>  |
| ssh.agent                           | Use a (local) ssh-agent.   | boolean   | <i>false</i> |
| ssh.agentForwarding                 | Use ssh-agent forwarding   | boolean   | <i>false</i> |
| ssh.timeout                         | The timeout for the connection setup and authentication (in milliseconds). | long      | <i>10000</i> |
| ssh.queue.pollingDelay              | The polling delay for monitoring running jobs (in milliseconds).           | long      | <i>1000</i>  |
| ssh.queue.multi.maxConcurrentJobs   | The maximum number of concurrent jobs in the multiq..                      | integer   | <i>4</i>     |
| local.queue.pollingDelay            | The polling delay for monitoring running jobs (in milliseconds).           | long      | <i>1000</i>  |
| local.queue.multi.maxConcurrentJobs | The maximum number of concurrent jobs in the multiq..                      | integer   | <i>4</i>     |



### 3.2.4 Slurm

The Slurm Adaptor submits jobs to a Slurm scheduler. This adaptor uses either the local or the ssh scheduler adaptor to run commands on the machine running Slurm, and the file or the stfp filesystem adaptor to gain access to the filesystem of that machine.

| field                 | value  |
|-----------------------|--|
| is_embedded           | False  |
| supports_interactive  | True   |
| supports_batch        | True   |
| uses_file_system      | True   |
| supported_credentials | <i>DefaultCredential, CertificateCredential, PasswordCredential, CredentialMap</i> |

**location string:**

- *local:///workdir]*
- *ssh://host[:port][workdir][ via:otherhost[:port]]\**

supported properties:

| name                                | description   | data_type | default      |
|-------------------------------------|---|-----------|--------------|
| slurm.disable.accounting.usage      | Do not use accounting info of slurm, even when available. Mostly for testing purposes | boolean   | <i>false</i> |
| slurm.poll.delay                    | Number of milliseconds between polling the status of a job.                           | long      | <i>1000</i>  |
| ssh.strictHostKeyChecking           | Enable strict host key checking.  | boolean   | <i>true</i>  |
| ssh.loadKnownHosts                  | Load the standard known_hosts file.   | boolean   | <i>true</i>  |
| ssh.loadSshConfig                   | Load the OpenSSH config file.   | boolean   | <i>true</i>  |
| ssh.agent                           | Use a (local) ssh-agent.  | boolean   | <i>false</i> |
| ssh.agentForwarding                 | Use ssh-agent forwarding  | boolean   | <i>false</i> |
| ssh.timeout                         | The timeout for the connection setup and authentication (in milliseconds).            | long      | <i>10000</i> |
| ssh.queue.pollingDelay              | The polling delay for monitoring running jobs (in milliseconds).                      | long      | <i>1000</i>  |
| ssh.queue.multi.maxConcurrentJobs   | The maximum number of concurrent jobs in the multiq..                                 | integer   | <i>4</i>     |
| local.queue.pollingDelay            | The polling delay for monitoring running jobs (in milliseconds).                      | long      | <i>1000</i>  |
| local.queue.multi.maxConcurrentJobs | The maximum number of concurrent jobs in the multiq.                                  | integer   | <i>4</i>     |

### 3.2.5 Gridengine

The SGE Adaptor submits jobs to a (Sun/Oracle/Univa) Grid Engine scheduler. This adaptor uses either the local or the ssh scheduler adaptor to run commands on the machine running Grid Engine, and the file or the stfp filesystem adaptor to gain access to the filesystem of that machine.

| field                 | value  |
|-----------------------|--|
| is_embedded           | False  |
| supports_interactive  | False  |
| supports_batch        | True   |
| uses_file_system      | True   |
| supported_credentials | <i>DefaultCredential, CertificateCredential, PasswordCredential, CredentialMap</i> |

**location string:**

- `local:///[/workdir]`
- `ssh://host[:port][[/workdir]][ via:otherhost[:port]]*`

supported properties:

| name  | description  | data_type | default      |
|---|--|-----------|--------------|
| <code>gri-dengine.ignore.version</code>           | Skip version check is skipped when connecting to remote machines. WARNING: it is not recommended to use this setting in production environments! | boolean   | <i>false</i> |
| <code>gri-dengine.accounting.grace.timeout</code> | Number of milliseconds a job is allowed to take going from the queue to the queue output.  | long      | <i>60000</i> |
| <code>gri-dengine.poll.delay</code>               | Number of milliseconds between polling the status of a job.  | long      | <i>1000</i>  |
| <code>ssh.strictHostKeyChecking</code>            | Enable strict host key checking.   | boolean   | <i>true</i>  |
| <code>ssh.loadKnownHosts</code>                   | Load the standard known_hosts file.  | boolean   | <i>true</i>  |
| <code>ssh.loadSshConfig</code>                    | Load the OpenSSH config file.  | boolean   | <i>true</i>  |
| <code>ssh.agent</code>                            | Use a (local) ssh-agent.   | boolean   | <i>false</i> |
| <code>ssh.agentForwarding</code>                  | Use ssh-agent forwarding   | boolean   | <i>false</i> |
| <code>ssh.timeout</code>                          | The timeout for the connection setup and authentication (in milliseconds).   | long      | <i>10000</i> |
| <code>ssh.queue.pollingDelay</code>               | The polling delay for monitoring running jobs (in milliseconds).   | long      | <i>1000</i>  |
| <code>ssh.queue.multi.maxConcurrentJobs</code>    | The maximum number of concurrent jobs in the multiq..  | integer   | <i>4</i>     |
| <code>local.queue.pollingDelay</code>             | The polling delay for monitoring running jobs (in milliseconds).   | long      | <i>1000</i>  |
| <code>local.queue.multi.maxConcurrentJobs</code>  | The maximum number of concurrent jobs in the multiq.   | integer   | <i>4</i>     |

### 3.2.6 Torque

The Torque Adaptor submits jobs to a TORQUE batch system. This adaptor uses either the local or the ssh scheduler adaptor to run commands on the machine running TORQUE, and the file or the stfp filesystem adaptor to gain access to the filesystem of that machine.

| field                              | value  |
|------------------------------------|--|
| <code>is_embedded</code>           | False  |
| <code>supports_interactive</code>  | False  |
| <code>supports_batch</code>        | True   |
| <code>uses_file_system</code>      | True   |
| <code>supported_credentials</code> | <i>DefaultCredential, CertificateCredential, PasswordCredential, CredentialMap</i> |

**location string:**

- `local:///[/workdir]`
- `ssh://host[:port][[/workdir]][ via:otherhost[:port]]*`

supported properties:

| name                                | description  | data_type | default |
|-------------------------------------|--|-----------|---------|
| torque.ignore.version               | Skip version check is skipped when connecting to remote machines. WARNING: it is not recommended to use this setting in production environments! | boolean   | false   |
| torque.accounting.grace             | Number of milliseconds a job is allowed to take going from the queue to the accinfo output.  | long      | 60000   |
| torque.poll.delay                   | Number of milliseconds between polling the status of a job.  | long      | 1000    |
| ssh.strictHostKeyChecking           | Enable strict host key checking.   | boolean   | true    |
| ssh.loadKnownHosts                  | Load the standard known_hosts file.  | boolean   | true    |
| ssh.loadSshConfig                   | Load the OpenSSH config file.  | boolean   | true    |
| ssh.agent                           | Use a (local) ssh-agent.   | boolean   | false   |
| ssh.agentForwarding                 | Use ssh-agent forwarding   | boolean   | false   |
| ssh.timeout                         | The timeout for the connection setup and authentication (in milliseconds).   | long      | 10000   |
| ssh.queue.pollingDelay              | The polling delay for monitoring running jobs (in milliseconds).   | long      | 1000    |
| ssh.queue.multi.maxConcurrentJobs   | The maximum number of concurrent jobs in the multiq..  | integer   | 4       |
| local.queue.pollingDelay            | The polling delay for monitoring running jobs (in milliseconds).   | long      | 1000    |
| local.queue.multi.maxConcurrentJobs | The maximum number of concurrent jobs in the multiq.   | integer   | 4       |



### Contents

- *API*
  - *The Server*
  - *File Systems*
    - \* *Message classes*
  - *Schedulers*
    - \* *Message classes*
  - *Credentials*
  - *Exceptions*

## 4.1 The Server

`xenon.init` (*port=None, do\_not\_exit=False, disable\_tls=False, log\_level='WARNING'*)

Start the Xenon GRPC server on the specified port, or, if a service is already running on that port, connect to that.

If no port is given, a random port is selected. This means that, by default, every python instance will start its own instance of a xenon-grpc process.

### Parameters

- **port** – the port number
- **do\_not\_exit** – by default the GRPC server is shut down after Python exits (through the *atexit* module), setting this value to *True* will prevent that from happening.

## 4.2 File Systems

**class** `xenon.FileSystem` (*service, wrapped*)

The Xenon *FileSystem* subsystem.

**Variables** `id` (*string*) – id

**append\_to\_file** (*path, data\_stream*)

Open an existing file and return an `OutputStream` to append data to this file.

**cancel** (*copy\_operation=None*)

Cancel a copy operation.

**Parameters** `copy_operation` (*xenon.CopyOperation*) – copy\_operation

**close** ()

Close this filestem Any pending/running copy operations of this filesystem will be terminated Will also forget this filesystem

**copy** (*source=None, destination\_filesystem=None, destination=None, mode=None, recursive=None*)

Asynchronously Copy an existing source path to a target path on a different file system.

**Parameters**

- **source** (*xenon.Path*) – source
- **destination\_filesystem** (*xenon.FileSystem*) – destination\_filesystem
- **destination** (*xenon.Path*) – destination
- **mode** (*xenon.CopyRequest.CopyMode*) – mode
- **recursive** (*bool*) – recursive

**classmethod** **create** (*adaptor=None, location=None, properties=None, certificate\_credential=None, password\_credential=None, default\_credential=None, credential\_map=None, keytab\_credential=None*)

Create a new `FileSystem` using the adaptor that connects to a data store at location using the credentials to get access.

**Parameters**

- **adaptor** (*string*) – adaptor
- **location** (*string*) – location
- **properties** (*map<string, string>*) – properties
- **certificate\_credential** (*xenon.CertificateCredential*) – certificate\_credential
- **password\_credential** (*xenon.PasswordCredential*) – password\_credential
- **default\_credential** (*xenon.DefaultCredential*) – default\_credential
- **credential\_map** (*xenon.CredentialMap*) – credential\_map
- **keytab\_credential** (*xenon.KeytabCredential*) – keytab\_credential

**create\_directories** (*path=None*)

Creates a new directory, including parent directories, failing if the directory already exists.

**Parameters** `path` (*xenon.Path*) – path

**create\_directory** (*path=None*)

Creates a new directory, failing if the directory already exists.

**Parameters** `path` (`xenon.Path`) – path

**create\_file** (`path=None`)

Creates a new empty file, failing if the file already exists.

**Parameters** `path` (`xenon.Path`) – path

**create\_symbolic\_link** (`link=None, target=None`)

Creates a new symbolic link, failing if the link already exists

**Parameters**

- **link** (`xenon.Path`) – link
- **target** (`xenon.Path`) – target

**delete** (`path=None, recursive=None`)

Deletes an existing path.

**Parameters**

- **path** (`xenon.Path`) – path
- **recursive** (`bool`) – recursive

**exists** (`path=None`)

Tests if a path exists.

**Parameters** `path` (`xenon.Path`) – path

**classmethod get\_adaptor\_description** (`name=None`)

Gives the description of the adaptor with the given name.

**Parameters** `name` (`string`) – name

**classmethod get\_adaptor\_descriptions** ()

Gives a list of the descriptions of the available adaptors.

**get\_adaptor\_name** ()

Get the name of the adaptor that created this FileSystem.

**classmethod get\_adaptor\_names** ()

Gives a list names of the available adaptors.

**get\_attributes** (`path=None`)

Get the PathAttributes of an existing path.

**Parameters** `path` (`xenon.Path`) – path

**get\_path\_separator** ()

Get the path separator used by this file system.

**get\_status** (`copy_operation=None`)

Retrieve the status of an copy.

**Parameters** `copy_operation` (`xenon.CopyOperation`) – copy\_operation

**get\_working\_directory** ()

Get the current working directory of this file system.

**is\_open** ()

Return if the connection to the FileSystem is open.

**list** (`dir=None, recursive=None`)

List all entries in the directory dir.

**Parameters**

- **dir** (`xenon.Path`) – dir
- **recursive** (`bool`) – recursive

**classmethod list\_file\_systems** ()

List the created filesystems Specific to grpc, not part of Xenon library

**classmethod local\_file\_systems** ()

Returns filesystems for all local drives Not part of FileSystem class in Xenon library In Xenon library available as LocalFileSystemUtils.getLocalFileSystems()

**read\_from\_file** (*path=None*)

Open an existing file and return an InputStream to read from this file.

**Parameters** **path** (`xenon.Path`) – path

**read\_symbolic\_link** (*path=None*)

Reads the target of a symbolic link

**Parameters** **path** (`xenon.Path`) – path

**rename** (*source=None, target=None*)

Rename an existing source path to a non-existing target path

**Parameters**

- **source** (`xenon.Path`) – source
- **target** (`xenon.Path`) – target

**set\_posix\_file\_permissions** (*path=None, permissions=None*)

Sets the POSIX permissions of a path

**Parameters**

- **path** (`xenon.Path`) – path
- **permissions** (`xenon.PosixFilePermission`) – permissions

**set\_working\_directory** (*path=None*)

Set the current working directory of this file system to directory.

**Parameters** **path** (`xenon.Path`) – path

**wait\_until\_done** (*copy\_operation=None, timeout=None*)

Wait until a copy operation is done or until a timeout expires.

**Parameters**

- **copy\_operation** (`xenon.CopyOperation`) – copy\_operation
- **timeout** (`uint64`) – timeout

**write\_to\_file** (*path, data\_stream*)

Open a file and return an OutputStream to write to this file. In Xenon library if request is missing size field then FileSystem.writeToFile(Path file) is used else FileSystem.writeToFile(Path path, long size) is used

**class** `xenon.Path` (*path*)

Wrapper around PurePosixPath from the pathlib module. This class reveals a string representation of the underlying path object to GRPC. You may use this class like a `pathlib.PurePosixPath`, including using it as an argument to `open` calls as it derives from `os.PathLike` (Python > 3.6). For more information see [the Python documentation on pathlib](#).

**is\_hidden** ()

Checks if a file is hidden. Just compares the first character in the filename with '.'.



### 4.2.1 Message classes

**class** `xenon.PosixFilePermission`

An enumeration.

`GROUP_EXECUTE = 6`

`GROUP_READ = 4`

`GROUP_WRITE = 5`

`NONE = 0`

`OTHERS_EXECUTE = 9`

`OTHERS_READ = 7`

`OTHERS_WRITE = 8`

`OWNER_EXECUTE = 3`

`OWNER_READ = 1`

`OWNER_WRITE = 2`

**class** `xenon.CopyMode`

An enumeration.

`CREATE = 0`

`IGNORE = 2`

`REPLACE = 1`

**class** `xenon.CopyStatus` (*service, wrapped*)

Status of a copy operation.

#### Variables

- `copy_operation` (`xenon.CopyOperation`) – copy\_operation
- `bytes_copied` (`uint64`) – bytes\_copied
- `bytes_to_copy` (`uint64`) – bytes\_to\_copy
- `done` (`bool`) – done
- `running` (`bool`) – running
- `state` (`string`) – state
- `error_message` (`string`) – error\_message
- `error_type` (`xenon.CopyStatus.ErrorType`) – error\_type

**class** `ErrorType`

An enumeration.

`ALREADY_EXISTS = 3`

`CANCELLED = 2`

`NONE = 0`

`NOT_CONNECTED = 4`

`NOT_FOUND = 1`

`XENON = 5`

`error_type`

## 4.3 Schedulers

**class** `xenon.Scheduler` (*service, wrapped*)

The Xenon Schedulers subsystem.

**Variables** `id` (*string*) – id

**cancel\_job** (*job=None*)

Cancel a job

**Parameters** `job` (`xenon.Job`) – job

**close** ()

Close this Scheduler. If scheduler is embedded then any pending/running jobs will be killed Will also forget this scheduler

**classmethod** `create` (*adaptor=None, location=None, properties=None, certificate\_credential=None, password\_credential=None, default\_credential=None, credential\_map=None, keytab\_credential=None*)

Create a new Scheduler using the adaptor connecting to the location using credentials to get access.

**Parameters**

- **adaptor** (*string*) – adaptor
- **location** (*string*) – location
- **properties** (*map<string, string>*) – properties
- **certificate\_credential** (`xenon.CertificateCredential`) – certificate\_credential
- **password\_credential** (`xenon.PasswordCredential`) – password\_credential
- **default\_credential** (`xenon.DefaultCredential`) – default\_credential
- **credential\_map** (`xenon.CredentialMap`) – credential\_map
- **keytab\_credential** (`xenon.KeytabCredential`) – keytab\_credential

**classmethod** `get_adaptor_description` (*name=None*)

Gives the description of the adaptor with the given name.

**Parameters** `name` (*string*) – name

**classmethod** `get_adaptor_descriptions` ()

Gives a list of the descriptions of the available adaptors.

**get\_adaptor\_name** ()

Get the name of the adaptor that created this Scheduler.

**classmethod** `get_adaptor_names` ()

Gives a list names of the available adaptors.

**get\_default\_queue\_name** ()

Get the name of the default queue.

**get\_file\_system** ()

Retrieve the FileSystem used internally by this Scheduler.

**get\_job\_status** (*job=None*)

Get the status of a Job.

**Parameters** **job** (*xenon.Job*) – job

**get\_job\_statuses** (*jobs=None*)

Get the status of all specified jobs.

**Parameters** **jobs** (*xenon.Job*) – jobs

**get\_jobs** (*queues=None*)

Get all job identifier of jobs currently in (one ore more) queues.

**Parameters** **queues** (*string*) – queues

**get\_location** ()

Get the location that this Scheduler is connected to.

**get\_properties** ()

Get the properties used to create this Scheduler.

**get\_queue\_names** ()

Get the queue names supported by this Scheduler.

**get\_queue\_status** (*queue=None*)

Get the status of the queue.

**Parameters** **queue** (*string*) – queue

**get\_queue\_statuses** (*queues=None*)

Get the status of all queues.

**Parameters** **queues** (*string*) – queues

**is\_open** ()

Test if the connection of this Scheduler is open.

**classmethod list\_schedulers** ()

List the created schedulers Specific to grpc, not part of Xenon library

**classmethod local\_scheduler** ()

Get scheduler on local filesystem with default location, credential and no properties Not part of Scheduler class in Xenon library In Xenon library available as Scheduler.create(“local”)

**submit\_batch\_job** (*description=None*)

Submit a batch job.

**Parameters** **description** (*xenon.JobDescription*) – description

**submit\_interactive\_job** (*description, stdin\_stream*)

Submit an interactive job The first response message in the response stream will contain the job identifier and empty stdout and stderr. Other response messages will also contain the job identifier and filled stdout and/or stderr.

**wait\_until\_done** (*job=None, timeout=None*)

Wait until a job is done or until a timeout expires.

**Parameters**

- **job** (*xenon.Job*) – job
- **timeout** (*uint64*) – timeout

**wait\_until\_running** (*job=None, timeout=None*)

Wait until a job starts running, or until a timeout expires.

**Parameters**

- **job** (`xenon.Job`) – job
- **timeout** (`uint64`) – timeout

### 4.3.1 Message classes

**class** `xenon.Job` (*id\_*)  
Job.

**Variables** **id** (*string*) – the Xenon job identifier.

**class** `xenon.JobDescription` (*\*\*kwargs*)  
This class describes a job to a Scheduler instance.

**Variables**

- **executable** (*string*) – executable
- **arguments** (*string*) – arguments
- **working\_directory** (*string*) – working\_directory
- **environment** (*map<string, string>*) – environment
- **queue\_name** (*string*) – queue\_name
- **max\_runtime** (*uint32*) – max\_runtime
- **stderr** (*string*) – stderr
- **stdin** (*string*) – stdin
- **stdout** (*string*) – stdout
- **name** (*string*) – name
- **max\_memory** (*uint32*) – max\_memory
- **scheduler\_arguments** (*string*) – scheduler\_arguments
- **tasks** (*uint32*) – tasks
- **cores\_per\_task** (*uint32*) – cores\_per\_task
- **tasks\_per\_node** (*uint32*) – tasks\_per\_node
- **start\_per\_task** (*bool*) – start\_per\_task
- **start\_time** (*string*) – start\_time
- **temp\_space** (*uint32*) – temp\_space

**class** `xenon.JobStatus` (*service, wrapped*)  
Status of a job.

**Variables**

- **job** (`xenon.Job`) – job
- **state** (*string*) – state
- **running** (*bool*) – running
- **done** (*bool*) – done

- **scheduler\_specific\_information** (*map<string, string>*) – scheduler\_specific\_information
- **exit\_code** (*int32*) – exit\_code
- **error\_message** (*string*) – error\_message
- **error\_type** (*xenon.JobStatus.ErrorType*) – error\_type
- **name** (*string*) – name

**class ErrorType**

An enumeration.

**CANCELLED** = 2

**IO** = 5

**NONE** = 0

**NOT\_CONNECTED** = 3

**NOT\_FOUND** = 1

**OTHER** = 6

**XENON** = 4

**error\_type**

**class xenon.QueueStatus** (*service, wrapped*)

Status of a queue.

**Variables**

- **name** (*string*) – name
- **scheduler\_specific\_information** (*map<string, string>*) – scheduler\_specific\_information
- **error\_message** (*string*) – error\_message
- **error\_type** (*xenon.QueueStatus.ErrorType*) – error\_type

**class ErrorType**

An enumeration.

**IO** = 4

**NONE** = 0

**NOT\_CONNECTED** = 2

**NOT\_FOUND** = 1

**OTHER** = 5

**XENON** = 3

**error\_type**

## 4.4 Credentials

**class xenon.CertificateCredential**

**certfile**

Field xenon.CertificateCredential.certfile

**passphrase**

Field xenon.CertificateCredential.passphrase

**username**

Field xenon.CertificateCredential.username

**class xenon.PasswordCredential**

**password**

Field xenon.PasswordCredential.password

**username**

Field xenon.PasswordCredential.username

## 4.5 Exceptions

**exception** xenon.exceptions.**AttributeNotSupportedException** (*method*, *exc\_code*,  
*exc\_msg*)

TODO: add doc-string.

**exception** xenon.exceptions.**CopyCancelledException** (*method*, *exc\_code*, *exc\_msg*)

TODO: add doc-string.

**exception** xenon.exceptions.**DirectoryNotEmptyException** (*method*, *exc\_code*, *exc\_msg*)

TODO: add doc-string.

**exception** xenon.exceptions.**FileSystemClosedException** (*method*, *exc\_code*, *exc\_msg*)

TODO: add doc-string.

**exception** xenon.exceptions.**IncompleteJobDescriptionException** (*method*, *exc\_code*,  
*exc\_msg*)

TODO: add doc-string.

**exception** xenon.exceptions.**InvalidCredentialException** (*method*, *exc\_code*, *exc\_msg*)

TODO: add doc-string.

**exception** xenon.exceptions.**InvalidJobDescriptionException** (*method*, *exc\_code*,  
*exc\_msg*)

TODO: add doc-string.

**exception** xenon.exceptions.**InvalidLocationException** (*method*, *exc\_code*, *exc\_msg*)

TODO: add doc-string.

**exception** xenon.exceptions.**InvalidOptionsException** (*method*, *exc\_code*, *exc\_msg*)

TODO: add doc-string.

**exception** xenon.exceptions.**InvalidPathException** (*method*, *exc\_code*, *exc\_msg*)

TODO: add doc-string.

**exception** xenon.exceptions.**InvalidPropertyException** (*method*, *exc\_code*, *exc\_msg*)

TODO: add doc-string.

**exception** xenon.exceptions.**InvalidResumeTargetException** (*method*, *exc\_code*,  
*exc\_msg*)

TODO: add doc-string.

```

exception xenon.exceptions.NoSuchCopyException(method, exc_code, exc_msg)
    TODO: add doc-string.

exception xenon.exceptions.NoSuchJobException(method, exc_code, exc_msg)
    TODO: add doc-string.

exception xenon.exceptions.NoSuchPathException(method, exc_code, exc_msg)
    TODO: add doc-string.

exception xenon.exceptions.NoSuchQueueException(method, exc_code, exc_msg)
    TODO: add doc-string.

exception xenon.exceptions.PathAlreadyExistsException(method, exc_code, exc_msg)
    Exception that is raised if FileSystem.create_directory() fails due to an existing path.

exception xenon.exceptions.PropertyTypeException(method, exc_code, exc_msg)
    TODO: add doc-string.

exception xenon.exceptions.UnknownAdaptorException(method, exc_code, exc_msg)
    TODO: add doc-string.

exception xenon.exceptions.UnknownPropertyException(method, exc_code, exc_msg)
    TODO: add doc-string.

exception xenon.exceptions.UnknownRpcException(method, exc_code, exc_msg)
    Default exception if nothing is known.

exception xenon.exceptions.UnsupportedJobDescriptionException(method,
                                                                exc_code,
                                                                exc_msg)
    TODO: add doc-string.

exception xenon.exceptions.UnsupportedOperationException(method,      exc_code,
                                                                exc_msg)
    TODO: add doc-string.

exception xenon.exceptions.XenonException(method, code, msg)
    Xenon base exception.

exception xenon.exceptions.XenonRuntimeException(method, exc_code, exc_msg)
    TODO: add doc-string.

xenon.exceptions.make_exception(method, e)
    Creates an exception for a given method, and RpcError.

```

The PyXenon module interfaces with Xenon-GRPC to get an interface to the Xenon 2.0 Java library. We kept this interface close to the original Java API. PyXenon 2.0 only works on Python 3.





## CHAPTER 5

---

### Installing

---

```
pip install pyxenon
```



## CHAPTER 6

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`



### X

`xenon`, [17](#)

`xenon.exceptions`, [26](#)



## A

ALREADY\_EXISTS (*xenon.CopyStatus.ErrorType* attribute), 21  
 append\_to\_file() (*xenon.FileSystem* method), 18  
 AttributeNotSupportedException, 26

## C

cancel() (*xenon.FileSystem* method), 18  
 cancel\_job() (*xenon.Scheduler* method), 22  
 CANCELLED (*xenon.CopyStatus.ErrorType* attribute), 21  
 CANCELLED (*xenon.JobStatus.ErrorType* attribute), 25  
 certfile (*xenon.CertificateCredential* attribute), 25  
 CertificateCredential (*class in xenon*), 25  
 close() (*xenon.FileSystem* method), 18  
 close() (*xenon.Scheduler* method), 22  
 copy() (*xenon.FileSystem* method), 18  
 CopyCancelledException, 26  
 CopyMode (*class in xenon*), 21  
 CopyStatus (*class in xenon*), 21  
 CopyStatus.ErrorType (*class in xenon*), 21  
 CREATE (*xenon.CopyMode* attribute), 21  
 create() (*xenon.FileSystem* class method), 18  
 create() (*xenon.Scheduler* class method), 22  
 create\_directories() (*xenon.FileSystem* method), 18  
 create\_directory() (*xenon.FileSystem* method), 18  
 create\_file() (*xenon.FileSystem* method), 19  
 create\_symbolic\_link() (*xenon.FileSystem* method), 19

## D

delete() (*xenon.FileSystem* method), 19  
 DirectoryNotEmptyException, 26

## E

error\_type (*xenon.CopyStatus* attribute), 21  
 error\_type (*xenon.JobStatus* attribute), 25  
 error\_type (*xenon.QueueStatus* attribute), 25

exists() (*xenon.FileSystem* method), 19

## F

FileSystem (*class in xenon*), 18  
 FileSystemClosedException, 26

## G

get\_adaptor\_description() (*xenon.FileSystem* class method), 19  
 get\_adaptor\_description() (*xenon.Scheduler* class method), 22  
 get\_adaptor\_descriptions() (*xenon.FileSystem* class method), 19  
 get\_adaptor\_descriptions() (*xenon.Scheduler* class method), 22  
 get\_adaptor\_name() (*xenon.FileSystem* method), 19  
 get\_adaptor\_name() (*xenon.Scheduler* method), 22  
 get\_adaptor\_names() (*xenon.FileSystem* class method), 19  
 get\_adaptor\_names() (*xenon.Scheduler* class method), 22  
 get\_attributes() (*xenon.FileSystem* method), 19  
 get\_default\_queue\_name() (*xenon.Scheduler* method), 22  
 get\_file\_system() (*xenon.Scheduler* method), 22  
 get\_job\_status() (*xenon.Scheduler* method), 22  
 get\_job\_statuses() (*xenon.Scheduler* method), 23  
 get\_jobs() (*xenon.Scheduler* method), 23  
 get\_location() (*xenon.Scheduler* method), 23  
 get\_path\_separator() (*xenon.FileSystem* method), 19  
 get\_properties() (*xenon.Scheduler* method), 23  
 get\_queue\_names() (*xenon.Scheduler* method), 23  
 get\_queue\_status() (*xenon.Scheduler* method), 23  
 get\_queue\_statuses() (*xenon.Scheduler* method), 23  
 get\_status() (*xenon.FileSystem* method), 19  
 get\_working\_directory() (*xenon.FileSystem* method), 19

GROUP\_EXECUTE (*xenon.PosixFilePermission* attribute), 21  
GROUP\_READ (*xenon.PosixFilePermission* attribute), 21  
GROUP\_WRITE (*xenon.PosixFilePermission* attribute), 21

## I

IGNORE (*xenon.CopyMode* attribute), 21  
IncompleteJobDescriptionException, 26  
init() (in module *xenon*), 17  
InvalidCredentialException, 26  
InvalidJobDescriptionException, 26  
InvalidLocationException, 26  
InvalidOptionsException, 26  
InvalidPathException, 26  
InvalidPropertyException, 26  
InvalidResumeTargetException, 26  
IO (*xenon.JobStatus.ErrorType* attribute), 25  
IO (*xenon.QueueStatus.ErrorType* attribute), 25  
is\_hidden() (*xenon.Path* method), 20  
is\_open() (*xenon.FileSystem* method), 19  
is\_open() (*xenon.Scheduler* method), 23

## J

Job (class in *xenon*), 24  
JobDescription (class in *xenon*), 24  
JobStatus (class in *xenon*), 24  
JobStatus.ErrorType (class in *xenon*), 25

## L

list() (*xenon.FileSystem* method), 19  
list\_file\_systems() (*xenon.FileSystem* class method), 20  
list\_schedulers() (*xenon.Scheduler* class method), 23  
local\_file\_systems() (*xenon.FileSystem* class method), 20  
local\_scheduler() (*xenon.Scheduler* class method), 23

## M

make\_exception() (in module *xenon.exceptions*), 27

## N

NONE (*xenon.CopyStatus.ErrorType* attribute), 21  
NONE (*xenon.JobStatus.ErrorType* attribute), 25  
NONE (*xenon.PosixFilePermission* attribute), 21  
NONE (*xenon.QueueStatus.ErrorType* attribute), 25  
NoSuchCopyException, 26  
NoSuchJobException, 27  
NoSuchPathException, 27  
NoSuchQueueException, 27  
NOT\_CONNECTED (*xenon.CopyStatus.ErrorType* attribute), 21

NOT\_CONNECTED (*xenon.JobStatus.ErrorType* attribute), 25  
NOT\_CONNECTED (*xenon.QueueStatus.ErrorType* attribute), 25  
NOT\_FOUND (*xenon.CopyStatus.ErrorType* attribute), 21  
NOT\_FOUND (*xenon.JobStatus.ErrorType* attribute), 25  
NOT\_FOUND (*xenon.QueueStatus.ErrorType* attribute), 25

## O

OTHER (*xenon.JobStatus.ErrorType* attribute), 25  
OTHER (*xenon.QueueStatus.ErrorType* attribute), 25  
OTHERS\_EXECUTE (*xenon.PosixFilePermission* attribute), 21  
OTHERS\_READ (*xenon.PosixFilePermission* attribute), 21  
OTHERS\_WRITE (*xenon.PosixFilePermission* attribute), 21  
OWNER\_EXECUTE (*xenon.PosixFilePermission* attribute), 21  
OWNER\_READ (*xenon.PosixFilePermission* attribute), 21  
OWNER\_WRITE (*xenon.PosixFilePermission* attribute), 21

## P

passphrase (*xenon.CertificateCredential* attribute), 26  
password (*xenon.PasswordCredential* attribute), 26  
PasswordCredential (class in *xenon*), 26  
Path (class in *xenon*), 20  
PathAlreadyExistsException, 27  
PosixFilePermission (class in *xenon*), 21  
PropertyTypeException, 27

## Q

QueueStatus (class in *xenon*), 25  
QueueStatus.ErrorType (class in *xenon*), 25

## R

read\_from\_file() (*xenon.FileSystem* method), 20  
read\_symbolic\_link() (*xenon.FileSystem* method), 20  
rename() (*xenon.FileSystem* method), 20  
REPLACE (*xenon.CopyMode* attribute), 21

## S

Scheduler (class in *xenon*), 22  
set\_posix\_file\_permissions() (*xenon.FileSystem* method), 20  
set\_working\_directory() (*xenon.FileSystem* method), 20  
submit\_batch\_job() (*xenon.Scheduler* method), 23  
submit\_interactive\_job() (*xenon.Scheduler* method), 23



## U

`UnknownAdaptorException`, [27](#)  
`UnknownPropertyException`, [27](#)  
`UnknownRpcException`, [27](#)  
`UnsupportedJobDescriptionException`, [27](#)  
`UnsupportedOperationException`, [27](#)  
`username` (*xenon.CertificateCredential attribute*), [26](#)  
`username` (*xenon.PasswordCredential attribute*), [26](#)

## W

`wait_until_done()` (*xenon.FileSystem method*), [20](#)  
`wait_until_done()` (*xenon.Scheduler method*), [23](#)  
`wait_until_running()` (*xenon.Scheduler method*),  
[23](#)  
`write_to_file()` (*xenon.FileSystem method*), [20](#)

## X

`xenon` (*module*), [17](#)  
`XENON` (*xenon.CopyStatus.ErrorType attribute*), [21](#)  
`XENON` (*xenon.JobStatus.ErrorType attribute*), [25](#)  
`XENON` (*xenon.QueueStatus.ErrorType attribute*), [25](#)  
`xenon.exceptions` (*module*), [26](#)  
`XenonException`, [27](#)  
`XenonRuntimeException`, [27](#)