PYTHON S3 EXAMPLES

CREATING A CONNECTION

This creates a connection so that you can interact with the server.

```
import boto
import boto.s3.connection
access_key = 'put your access key here!'
secret_key = 'put your secret key here!'

conn = boto.connect_s3(
        aws_access_key_id = access_key,
        aws_secret_access_key = secret_key,
        host = 'objects.dreamhost.com',
        #is_secure=False,               # uncomment if you are not using ssl
        calling_format = boto.s3.connection.OrdinaryCallingFormat(),
        )
```

LISTING OWNED BUCKETS

This gets a list of Buckets that you own. This also prints out the bucket name and creation date of each bucket.

```
for bucket in conn.get_all_buckets():
        print "{name}\t{created}".format(
                name = bucket.name,
                created = bucket.creation_date,
        )
```

The output will look something like this:

```
mahbuckat1   2011-04-21T18:05:39.000Z
mahbuckat2   2011-04-21T18:05:48.000Z
mahbuckat3   2011-04-21T18:07:18.000Z
```

CREATING A BUCKET

This creates a new bucket called `my-new-bucket`

```
bucket = conn.create_bucket('my-new-bucket')
```

LISTING A BUCKET'S CONTENT

This gets a list of objects in the bucket. This also prints out each object's name, the file size, and last modified date.

```
for key in bucket.list():
        print "{name}\t{size}\t{modified}".format(
                name = key.name,
                size = key.size,
                modified = key.last_modified,
                )
```

The output will look something like this:

```
myphoto1.jpg 251262  2011-08-08T21:35:48.000Z
```

```
myphoto2.jpg 262518  2011-08-08T21:38:01.000Z
```

DELETING A BUCKET

Note: The Bucket must be empty! Otherwise it won't work!

```
conn.delete_bucket(bucket.name)
```

FORCED DELETE FOR NON-EMPTY BUCKETS

Attention: not available in python

CREATING AN OBJECT

This creates a file `hello.txt` with the string `"Hello World!"`

```
key = bucket.new_key('hello.txt')
key.set_contents_from_string('Hello World!')
```

CHANGE AN OBJECT'S ACL

This makes the object `hello.txt` to be publicly readable, and `secret_plans.txt` to be private.

```
hello_key = bucket.get_key('hello.txt')
hello_key.set_canned_acl('public-read')
plans_key = bucket.get_key('secret_plans.txt')
plans_key.set_canned_acl('private')
```

DOWNLOAD AN OBJECT (TO A FILE)

This downloads the object `perl_poetry.pdf` and saves it in `/home/larry/documents/`

```
key = bucket.get_key('perl_poetry.pdf')
key.get_contents_to_filename('/home/larry/documents/perl_poetry.pdf')
```

DELETE AN OBJECT

This deletes the object `goodbye.txt`

```
bucket.delete_key('goodbye.txt')
```

GENERATE OBJECT DOWNLOAD URLS (SIGNED AND UNSIGNED)

This generates an unsigned download URL for `hello.txt`. This works because we made `hello.txt` public by setting the ACL above. This then generates a signed download URL for `secret_plans.txt` that will work for 1 hour. Signed download URLs will work for the time period even if the object is private (when the time period is up, the URL will stop working).

```
hello_key = bucket.get_key('hello.txt')
hello_url = hello_key.generate_url(0, query_auth=False, force_http=True)
print hello_url
```

```
plans_key = bucket.get_key('secret_plans.txt')
plans_url = plans_key.generate_url(3600, query_auth=True, force_http=True)
print plans_url
```

The output of this will look something like:

```
http://objects.dreamhost.com/my-bucket-name/hello.txt
http://objects.dreamhost.com/my-bucket-name/secret_plans.txt?Signature=XXXXXXXXXXXXXXXXXXXXXXX
```