2.7. 网格文件系统

2.7.1. 写作

GridFS 支持以字段对象的形式出现`FileField`。该字段充当类似文件的对象，并提供几种不同的插入和检索数据的方法。诸如内容类型之类的任意元数据也可以与文件一起存储。访问 FileField 时返回的对象是Pymongo 的 GridFS 的代理 在以下示例中，创建了一个文档来存储有关动物的详细信息，包括照片：

```
class Animal(Document):
    genus = StringField()
    family = StringField()
    photo = FileField()

marmot = Animal(genus='Marmota', family='Sciuridae')

with open('marmot.jpg', 'rb') as fd:
    marmot.photo.put(fd, content_type = 'image/jpeg')
marmot.save()
```

2.7.2. 恢复

所以使用 the`FileField`就像使用任何其他字段一样。该文件也可以很容易地检索：

```
marmot = Animal.objects(genus='Marmota').first()
photo = marmot.photo.read()
content_type = marmot.photo.content_type
```

笔记

如果您需要多次 read() 文件的内容，则需要使用*seek*来“倒回”类文件对象：

```
marmot = Animal.objects(genus='Marmota').first()
content1 = marmot.photo.read()
assert content1 != ""

content2 = marmot.photo.read()    # will be empty
assert content2 == ""

marmot.photo.seek(0)              # rewind the file by setting the current position
of the cursor in the file to 0
content3 = marmot.photo.read()
assert content3 == content1
```

2.7.3. 串流

将数据流式传输到 a 的 `FileField` 方式略有不同。首先，必须通过调用该方法创建一个新文件 `new_file()`。然后可以使用以下方式写入数据 `write()`：

```
marmot.photo.new_file()
marmot.photo.write('some_image_data')
marmot.photo.write('some_more_image_data')
marmot.photo.close()

marmot.save()
```

2.7.4. 删除

删除存储的文件是通过以下 `delete()` 方法实现的：

```
marmot.photo.delete()    # Deletes the GridFS document
marmot.save()            # Saves the GridFS reference (being None) contained in the
marmot instance
```

警告

Document 中的 FileField 实际上仅将文件的 ID 存储在单独的 GridFS 集合中。这意味着删除具有定义的 FileField 的文档实际上并不会删除该文件。在删除文档本身之前，您必须小心删除文档中的任何文件，如上所示。

2.7.5. 替换文件

可以使用该方法替换文件 `replace()`。这就像方法一样工作 `put()`，因此甚至可以（并且应该）替换元数据：

```
another_marmot = open('another_marmot.png', 'rb')
marmot.photo.replace(another_marmot, content_type='image/png')  # Replaces the GridFS
document
marmot.save()                                                   # Replaces the GridFS
reference contained in marmot instance
```