创建Cinder-API的扩展资源与核心资源扩展

扩展资源与核心资源扩展的python文件定义在 /cinder/api/contrib/ 目录下。在进行此类API扩展时，只要在该目录下放好python文件就可以添加一个自定义的API接口（通过追加配置项CONF. osapi\_volume\_extension可以将python文件放到其他路径）。

扩展资源：

扩展资源本身也是一种资源，只是没那么核心，比如定义storage资源。

其访问方式为：PUT /v2/< tenant\_id >/storage /<storage\_id>。

在OpenStack中一种资源表示对一类实体进行操作，如volume，snapshot，type等，因此创建一类新实体的API接口时可以创建一个扩展资源。

核心资源的扩展：

核心资源的扩展(Resource extension)，包括对 action 的扩展和基本操作（如GET）的扩展。如Cinder原有的核心资源VolumeController，在某种场景下需要对该资源添加一种新的操作，此时可以扩展VolumeController的Post /v2/< tenant\_id >/volume /<volume\_id>/action操作，让action多出一个响应如”os\_ruijie-new-action”。

扩展方法：

1. 在/cinder/api/contrib/下创建xxx.py
2. xxx.py中定义XXXController方法
3. xxx.py中定义扩展描述类（ExtensionDescriptor类的派生类），注意扩展描述类的类名是xxx.py文件名的首字母大写（不这样命名的话加载会失败）
4. 如果定义扩展资源，为扩展描述类实现get\_controller\_extensions接口；如果定义核心资源扩展，为扩展描述类实现get\_resources接口

范例：

扩展资源

|  |
| --- |
| *# -\*- coding:utf-8 -\*- from oslo\_log import log as logging import webob.exc from cinder.api import extensions from cinder.api.openstack import wsgi LOG = logging.getLogger(\_\_name\_\_)  class RuijieController(wsgi.Controller):  def \_\_init\_\_(self, ext\_mgr):  self.ext\_mgr = ext\_mgr  super(RuijieController, self).\_\_init\_\_()   def index(self, req):  """对应GET /v2/< tenant\_id >/ruijie/"""  LOG.debug("GET Request : {0}".format(req))  return webob.Response(status\_int=202)  class Ruijie\_demo(extensions.ExtensionDescriptor):  # 新增的扩展资源名称  name = "Rujie\_demo"  # 扩展资源名称的路由信息  # 路由时通过/v2/< tenant\_id >/ruijie/映射到RuijieController  alias = "os-ruijie"  updated = "2017-07-17T00:00:00-00:00"   def \_\_init\_\_(self,ext\_mgr):  self.\_\_doc\_\_ = "Demo"  super(Ruijie\_demo,self).\_\_init\_\_(ext\_mgr=ext\_mgr)   def get\_resources(self):  """重写get\_resources表示改资源是一个扩展资源"""  resources = []  controller = RuijieController(self.ext\_mgr)  resource = extensions.ResourceExtension('os-ruijie', controller)  resources.append(resource)  return resources* |

核心资源扩展：

扩展volume的action

|  |
| --- |
| *# -\*- coding:utf-8 -\*- from oslo\_log import log as logging import webob.exc from cinder.api import extensions from cinder.api.openstack import wsgi LOG = logging.getLogger(\_\_name\_\_)  class RuijieController(wsgi.Controller):   def \_\_init\_\_(self, \*args, \*\*kwargs):  super(RuijieController, self).\_\_init\_\_(\*args, \*\*kwargs)   @wsgi.action('os-ruijie\_demo')  def \_ruijie\_demo(self, req, id, body):  """  对应的URL请求：  POST /v2/< tenant\_id >/volumes/<volume\_id>/action  {  "os-ruijie\_demo":{}  }  """  LOG.debug("Request : {0}".format(req))  return webob.Response(status\_int=202)  class Ruijie\_demo(extensions.ExtensionDescriptor):  # 新增的扩展资源名称  name = "Rujie\_demo"  # 扩展资源名称的路由信息  # 路由时通过/v2/< tenant\_id >/ruijie/映射到RuijieController  alias = "os-ruijie"  updated = "2017-07-17T00:00:00-00:00"   def \_\_init\_\_(self,ext\_mgr):  self.\_\_doc\_\_ = "Demo"  super(Ruijie\_demo, self).\_\_init\_\_(ext\_mgr=ext\_mgr)   def get\_controller\_extensions(self):  exts = []  controller = RuijieController()  extension = extensions.ControllerExtension(self, "volumes", controller)  exts.append(extension)  return exts* |