#

# From oslo.cache

#

# Prefix for building the configuration dictionary for the cache region. This

# should not need to be changed unless there is another dogpile.cache region

# with the same configuration name. (string value)

#config\_prefix = cache.oslo

# Default TTL, in seconds, for any cached item in the dogpile.cache region.

# This applies to any cached method that doesn't have an explicit cache

# expiration time defined for it. (integer value)

#expiration\_time = 600

# Cache backend module. For eventlet-based or environments with hundreds of

# threaded servers, Memcache with pooling (oslo\_cache.memcache\_pool) is

# recommended. For environments with less than 100 threaded servers, Memcached

# (dogpile.cache.memcached) or Redis (dogpile.cache.redis) is recommended. Test

# environments with a single instance of the server can use the

# dogpile.cache.memory backend. (string value)

# Possible values:

# oslo\_cache.memcache\_pool - <No description provided>

# oslo\_cache.dict - <No description provided>

# oslo\_cache.mongo - <No description provided>

# oslo\_cache.etcd3gw - <No description provided>

# dogpile.cache.memcached - <No description provided>

# dogpile.cache.pylibmc - <No description provided>

# dogpile.cache.bmemcached - <No description provided>

# dogpile.cache.dbm - <No description provided>

# dogpile.cache.redis - <No description provided>

# dogpile.cache.memory - <No description provided>

# dogpile.cache.memory\_pickle - <No description provided>

# dogpile.cache.null - <No description provided>

#backend = dogpile.cache.null

backend = {{ \_data.get('backend', 'oslo\_cache.memcache\_pool') }}

# Arguments supplied to the backend module. Specify this option once per

# argument to be passed to the dogpile.cache backend. Example format:

# "<argname>:<value>". (multi valued)

#backend\_argument =

# Proxy classes to import that will affect the way the dogpile.cache backend

# functions. See the dogpile.cache documentation on changing-backend-behavior.

# (list value)

#proxies =

# Global toggle for caching. (boolean value)

#enabled = true

{%- if \_data.enabled is defined %}

{%- if \_data.enabled %}

enabled = True

{%- endif %}

{%- else %}

{%- if \_data.servers is defined or \_data.members is defined %}

enabled = True

{%- endif %}

{%- endif %}

# Extra debugging from the cache backend (cache keys, get/set/delete/etc

# calls). This is only really useful if you need to see the specific cache-

# backend get/set/delete calls with the keys/values. Typically this should be

# left set to false. (boolean value)

#debug\_cache\_backend = false

# Memcache servers in the format of "host:port". (dogpile.cache.memcache and

# oslo\_cache.memcache\_pool backends only). (list value)

#memcache\_servers = localhost:11211

{%- if \_data.servers is defined %}

memcached\_servers = {% for member,data in \_data.servers.iteritems() %}{% if data.get('enabled', False) %}{{ member }}:{{ data.port }}{% if not loop.last %},{% endif %}{% endif %}{%- endfor %}

{%- elif \_data.members is defined %}

memcache\_servers = {%- for member in \_data.members %}{{ member.host }}:{{ member.port }}{% if not loop.last %},{% endif %}{%- endfor %}

{%- endif %}

# Number of seconds memcached server is considered dead before it is tried

# again. (dogpile.cache.memcache and oslo\_cache.memcache\_pool backends only).

# (integer value)

#memcache\_dead\_retry = 300

# Timeout in seconds for every call to a server. (dogpile.cache.memcache and

# oslo\_cache.memcache\_pool backends only). (floating point value)

#memcache\_socket\_timeout = 3.0

# Max total number of open connections to every memcached server.

# (oslo\_cache.memcache\_pool backend only). (integer value)

#memcache\_pool\_maxsize = 10

# Number of seconds a connection to memcached is held unused in the pool before

# it is closed. (oslo\_cache.memcache\_pool backend only). (integer value)

#memcache\_pool\_unused\_timeout = 60

# Number of seconds that an operation will wait to get a memcache client

# connection. (integer value)

#memcache\_pool\_connection\_get\_timeout = 10