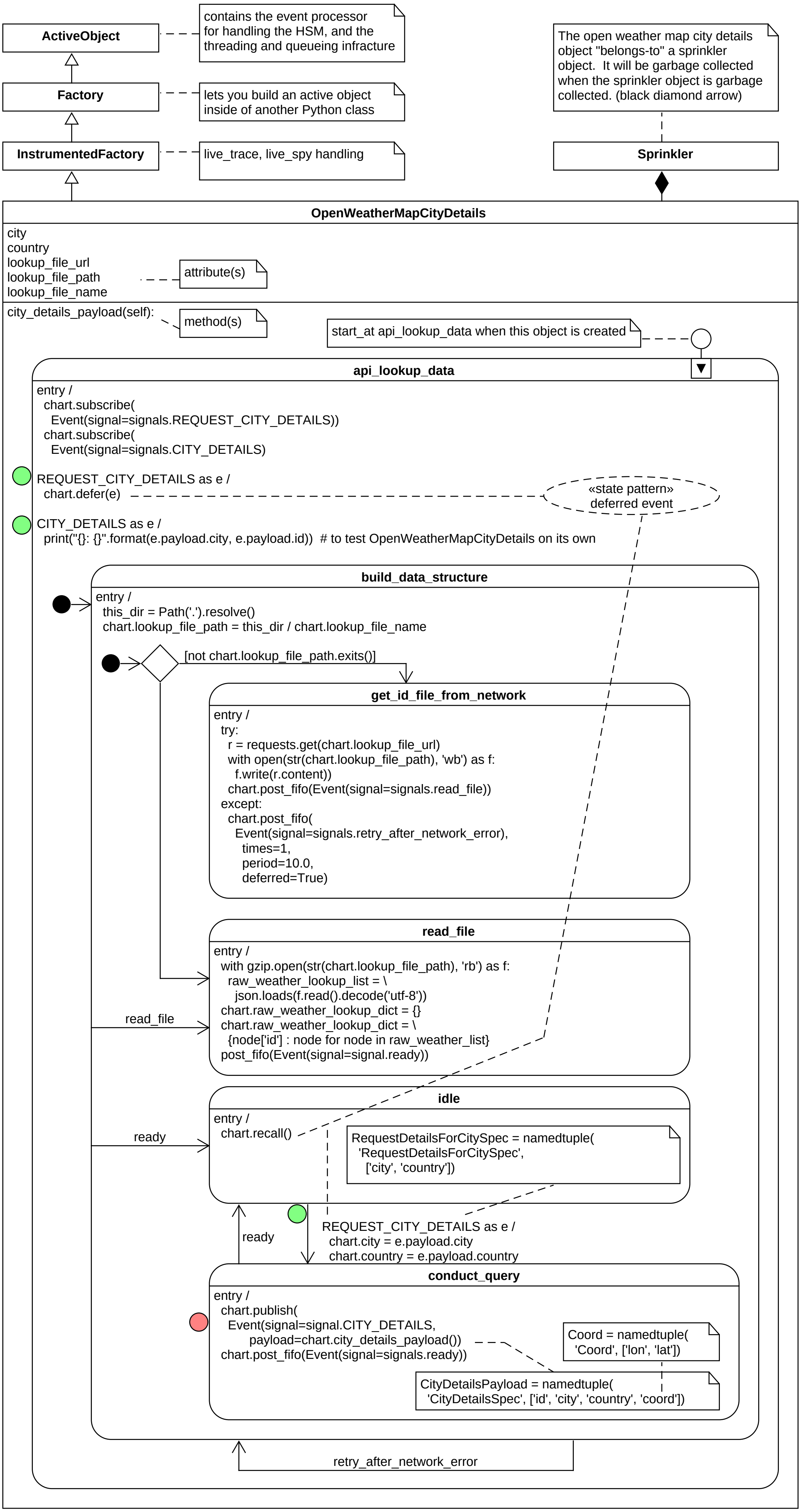
ActiveObject
contains the event processor for handling the HSM, and the threading and queueing infracture
Factory
lets you build an active object inside of another Python class
InstrumentedFactory
live_trace, live_spy handling
The open weather map city details object "belongs-to" a sprinkler object. It will be garbage collected when the sprinkler object is garbage collected. (black diamond arrow)
Sprinkler
OpenWeatherMapCityDetails
city
country
lookup_file_url
lookup_file_path
lookup_file_name
attribute(s)
city_details_payload(self):
method(s)
start_at api_lookup_data when this object is created
api_lookup_data
entry /
chart.subscribe(
Event(signal=signals.REQUEST_CITY_DETAILS))
chart.subscribe(
Event(signal=signals.CITY_DETAILS)
REQUEST_CITY_DETAILS as e /
chart.defer(e)
«state pattern»
deferred event
CITY_DETAILS as e /
print("{}: {}".format(e.payload.city, e.payload.id)) # to test OpenWeatherMapCityDetails on its own
build_data_structure
entry /
this_dir = Path('.').resolve()
chart.lookup_file_path = this_dir / chart.lookup_file_name
[not chart.lookup_file_path.exits()]
get_id_file_from_network
entry /
try:
r = requests.get(chart.lookup_file_url)
with open(str(chart.lookup_file_path), 'wb') as f:
f.write(r.content))
chart.post_fifo(Event(signal=signals.read_file))
except:
chart.post_fifo(
Event(signal=signals.retry_after_network_error),
times=1,
period=10.0,
deferred=True)
read_file
entry /
with gzip.open(str(chart.lookup_file_path), 'rb') as f:
raw_weather_lookup_list = \
json.loads(f.read().decode('utf-8'))
chart.raw_weather_lookup_dict = {}
chart.raw_weather_lookup_dict = \
{node['id'] : node for node in raw_weather_list}
post_fifo(Event(signal=signal.ready))
read_file
idle
entry /
chart.recall()
ready
RequestDetailsForCitySpec = namedtuple(
'RequestDetailsForCitySpec',
['city', 'country'])
ready
REQUEST_CITY_DETAILS as e /
chart.city = e.payload.city
chart.country = e.payload.country
conduct_query
entry /
chart.publish(
Event(signal=signal.CITY_DETAILS,
payload=chart.city_details_payload())
chart.post_fifo(Event(signal=signals.ready))
Coord = namedtuple(
'Coord', ['lon', 'lat'])
CityDetailsPayload = namedtuple(
'CityDetailsSpec', ['id', 'city', 'country', 'coord'])
retry_after_network_error