**1、OpenAI Gym是什么**

OpenAI Gym是一个用于**测试和比较**强化学习算法的工具包。强化学习需要提供一个环境给Agent运行，才能评测Agent的策略的优劣。OpenAI Gym 是**提供各种环境**的开源工具包。

1. **安装**
2. pip install gym（需要python3.5以上）

（2）使用git：

git clone https://github.com/openai/gymcd gym

pip install -e . **# minimal install**

pip install -e .[all] **# full install**

**3、OpenAI Gym组成**

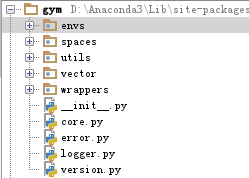
1.gym开源库：强化学习环境的集合。这些环境有一个公共的接口，允许用户设计通用的算法。

2.OpenAI Gym服务：提供一个站点和api[，允许用户对他们的测试结果进行比较。](https://gym.openai.com/envs/CartPole-v0）和api，允许用户对他们的测试结果进行比较。)https://gym.openai.com/envs/CartPole-v0

例如：CartPole问题，第一个算法在85个episodes后解决问题（当总奖励>195时认为解决问题）



1. **结构**



envs包括各种仿真环境，spaces包括各种空间数据结构的定义，utils是一些通用程序，wrappers用于将环境进行打包，可以使用这个模块来记录自己的算法在环境中的表现，并拍摄自己算法学习的视频。4个程序中，core.py 是核心程序包括很多基类，error.py是错误管理程序，logger.py控制打印信息，version.py记录当前版本。

3.1 envs

envs**类型**：

经典控制和文字游戏：经典的强化学习示例，方便入门；

算法：从例子中学习强化学习的相关算法，在 Gym 的仿真算法中，由易到难方便新手入坑；

雅达利游戏：利用强化学习来玩雅达利的游戏。Gym 中集成了对强化学习有着重要影响的 Arcade Learning Environment，并且方便用户安装；

2D 和 3D 的机器人：这个是我一直很感兴趣的一部分，在 Gym 中控制机器人进行仿真。需要利用第三方的物理引擎如 MuJoCo 。

env**核心方法:**

env.reset():重置环境的状态，返回一个初始observation

env.step(self,action):推进一个时间步长，返回observation，reward，done，info

env.render():重绘环境的一帧。默认模式一般比较友好，如弹出一个窗口。

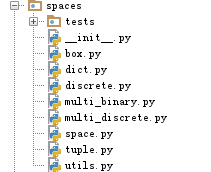


* Observation(object):返回一个特定环境的对象对环境的观察。比如，来自相机的像素数据，机器人的关节角度和关节速度，或棋盘游戏中的棋盘状态。
* Reward(float)：返回之前动作收获回报之和。不同的环境计算方式不一样，但总体的目标是增加总奖励。
* Done(boolean)：返回是否应该重新设置（reset）环境。done=true，表示当前episode结束。
* Info(dict)：用于调试的诊断信息。
* **原理：**
* 每一个时间步长，agent选择一个action执行,environment返回相应的observation和reward。

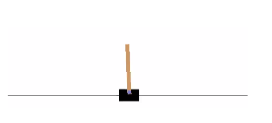


3.2 spaces

space：action space和observation space（描述有效的运动和观测的格式和范围)



3.1 例子：CartPole



**问题描述：**

操纵小车左右移动，使它上面的木棒能够保持平衡。杆开始直立，目标是通过增加和减少手推车的速度来防止它翻倒。

目标是坚持尽量长的时间，对于任意一个action，只要不导致任务失败，reward +1

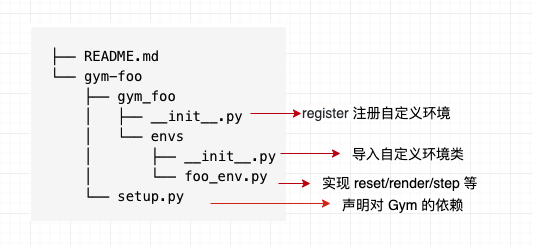
**问题定义：**  
Observation: Cart Position、Cart Velocity 、Pole Angle、Pole Velocity At Tip

observation\_space=box  
actions: 0、1（左、右） action\_space=discrete  
Reward: 每一个step,reward+1  
Starting State: observation所有随机赋值，范围 [-0.05，0.05]Episode Termination：Pole Angle超过12度；车位置距中点超过2.4单位长度；超过200个episode;Solved Requirements：当平均奖励大于或等于195.0时，视为解决

**利用gym编写强化学习环境**

自己构建的GYM环境是在envs下子文件夹中的一个py文件环境中的类：

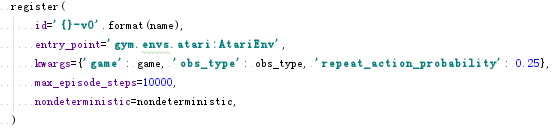
gym\envs\classic\_control\cartpole.py



**注册自定义环境**

构建的环境需要调用gym库，再导入gym库使用该环境。故需在GYM的内部创建一个内链接，指向自己构建的环境。

**1.**registry **(**envs\\_init\_.py**):**以ms\_pacman-v0为例



*Id:环境名 Entry\_point:所在位置*

*算法所需参数（obs\_type:image/ram nondeterministic=true/false）*

**2.**在相应文件夹下建立\_init\_.py：

**from** gym.envs.atari.atari\_env（文件名） **import** AtariEnv（类）

**GYM环境构建（环境类：含变量和函数）**

变量：state、action;对应的observation\_space、action\_space需要在类的init中定义

self.action\_space = spaces.Discrete(len(self.\_action\_set))

**if** self.\_obs\_type == **'ram'**:  
 self.observation\_space = spaces.Box(low=0, high=255, dtype=np.uint8, shape=(128,))  
**elif** self.\_obs\_type == **'image'**:  
 self.observation\_space = spaces.Box(low=0, high=255, shape=(screen\_height, screen\_width, 3), dtype=np.uint8)

取值在0-255维度128

函数：step(\*)、reset、render、

env.reset():重置环境的状态，返回一个初始observation

env.step(self,action):推进一个时间步长，返回observation，reward，done，info

env.render():重绘环境的一帧。默认模式一般比较友好，如弹出一个窗口。

敌方飞机：变换队形不断攻击，有生命值，阵亡后会有新飞机补位置

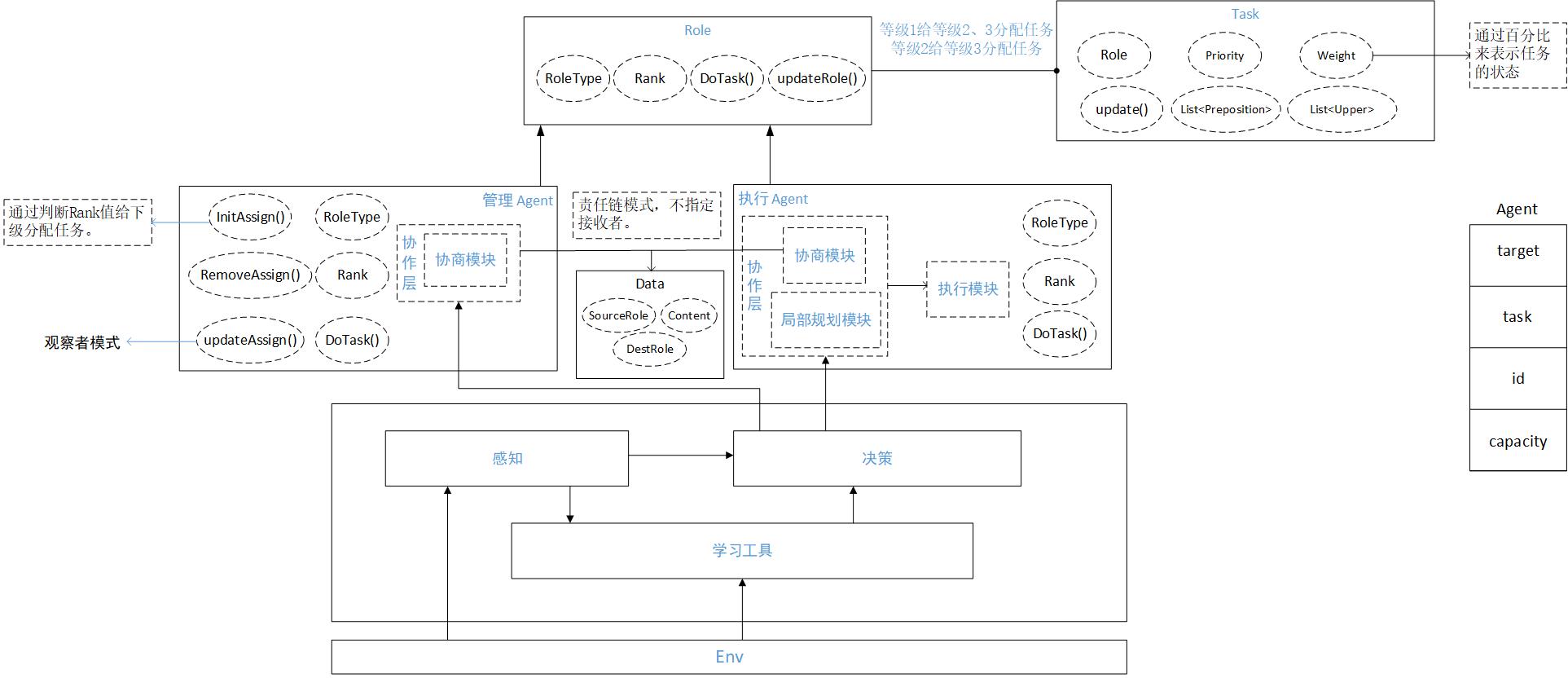
我方飞机：设定初始状态

**State**：用不同的图形表示我方飞机角色，图形变换对应角色切换，颜色变化表示任务完成度变化，有攻击状态和侦查状态，运输状态先不考虑（弹药用尽后自行停止一段时间补充，期间不能攻击）

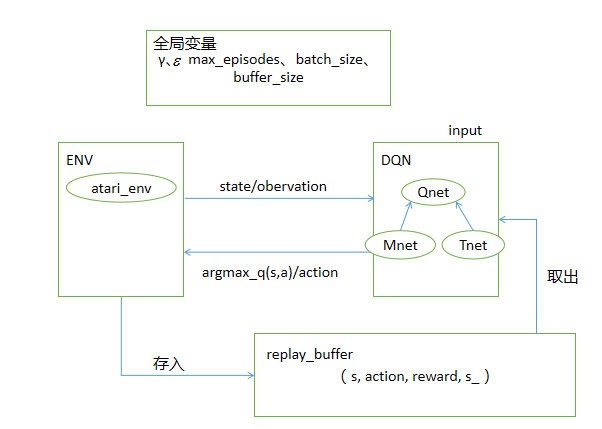
**Action**：攻击状态--fire、move（上下左右）；侦查状态--侦查（能获取侦查时间内敌机位置，不能攻击）；多个智能体，action类型是数组或者字典；

**Observation**：一帧图片，box类型

**Reward:** 击中敌机+，被击中-，消灭敌机++（每无损消耗敌机生命值弹药则认为消灭一架敌机），被击落--



**DQN**



输入：observation

输出：action