

# Gymnasium 环境介绍

[Gymnasium](#) (曾用名为 OpenAI Gym) 是由 OpenAI 提供的一个标准强化学习环境库，它为研究人员和开发者提供了一套统一的接口，用于创建和比较各种强化学习算法。Gymnasium 支持多种类型的环境，包括经典控制任务、离散和连续动作空间的任务、以及复杂的模拟环境。

如图 1 所示，对于每个环境，Gymnasium 简要介绍了其状态或观测 (Observation Space)、动作空间 (Action Space) 以及奖励机制 (Reward Mechanism)。这些信息对于理解环境的动态和设计合适的强化学习算法至关重要。



## Cart Pole



This environment is part of the Classic Control environments which contains general information about the environment.

Action Space	<code>Discrete(2)</code>
Observation Space	<code>Box([-4.8 -inf -0.41887903 -inf], [4.8 inf 0.41887903 inf], (4,), float32)</code>
import	<code>gymnasium.make("CartPole-v1")</code>

## Description

This environment corresponds to the version of the cart-pole problem described by Barto, Sutton, and Anderson in "[Neuronlike Adaptive Elements That Can Solve Difficult Learning Control Problem](#)". A pole is attached by an un-actuated joint to a cart, which moves along a frictionless track. The pendulum is placed upright on the cart and the goal is to balance the pole by applying forces in the left and right direction on the cart.

## Action Space

The action is a `ndarray` with shape `(1,)` which can take values `{0, 1}` indicating the direction of the fixed force the cart is pushed with.

- 0: Push cart to the left
- 1: Push cart to the right

图 1 Gymnasium 环境说明

在奖励说明中，通常会提到奖励临界值（Reward Threshold），这是一个预设的奖励水平，表示智能体在该环境中达到“成功”所需的最低奖励。例如，在某些环境中，达到特定的奖励临界值以上意味着智能体很有可能学会了完成任务的基本策略，具体还需可视化验证。

## Gymnasium 环境接口

在 Gymnasium 中，每个环境都遵循一个标准的接口，包括以下几个关键方法：

- `reset()`: 初始化环境并返回初始观测。
- `step(action)`: 执行动作并返回下一个观测、奖励、是否结束标志和额外信息。
- `render()`: 可视化当前环境状态。
- `close()`: 关闭环境并释放资源。

通过这些方法，用户可以轻松地与环境交互，收集数据，如代码 1 所示。

### 代码 1 Gymnasium 环境交互示例

```
import gymnasium as gym
# 创建环境
env = gym.make("CartPole-v1")
obs, info = env.reset() # 重置环境，获得初始观测或状态
for _ in range(100):
    # env.render() # 显示画面
    action = env.action_space.sample() # 随机采样一个动作
    obs, reward, done, truncated, info = env.step(action) # 与环境交互
    if done or truncated: # 如果回合结束，重置环境
        obs, info = env.reset()
env.close()
```

其中，`env.step(action)` 返回的 `done` 和 `truncated` 标志用于指示当前回合是否结束。`done` 通常表示智能体达到了终止状态，而 `truncated` 则表示由于时间限制等原因导致的回合结束，通常 `truncated` 会比 `done` 更早触发。