Install

In [5]:
```
import os
os.chdir('./GymGo-master')
```

Out[5]:
```
'/Users/yuetongliu/Desktop/Reinforcement_Learning_on_Chess/Code/14_02_24_Go/GymG
o-master'
```

In [6]:
```
! pip install -e .
```

```
Obtaining file:///Users/yuetongliu/Desktop/Reinforcement_Learning_on_Chess/Code/
14_02_24_Go/GymGo-master
Collecting gym
  Downloading gym-0.18.0.tar.gz (1.6 MB)
     |████████████████████████████████| 1.6 MB 3.3 MB/s eta 0:00:01
Requirement already satisfied: scipy in /opt/anaconda3/lib/python3.8/site-packag
es (from gym->gym-go==0.0.1) (1.5.2)
Requirement already satisfied: numpy>=1.10.4 in /opt/anaconda3/lib/python3.8/sit
e-packages (from gym->gym-go==0.0.1) (1.19.2)
Collecting pyglet<=1.5.0,>=1.4.0
  Downloading pyglet-1.5.0-py2.py3-none-any.whl (1.0 MB)
     |████████████████████████████████| 1.0 MB 9.2 MB/s eta 0:00:01
Collecting Pillow<=7.2.0
  Downloading Pillow-7.2.0-cp38-cp38-macosx_10_10_x86_64.whl (2.2 MB)
     |████████████████████████████████| 2.2 MB 15.9 MB/s eta 0:00:01
Requirement already satisfied: cloudpickle<1.7.0,>=1.2.0 in /opt/anaconda3/lib/p
ython3.8/site-packages (from gym->gym-go==0.0.1) (1.6.0)
Requirement already satisfied: future in /opt/anaconda3/lib/python3.8/site-packa
ges (from pyglet<=1.5.0,>=1.4.0->gym->gym-go==0.0.1) (0.18.2)
Building wheels for collected packages: gym
  Building wheel for gym (setup.py) ... done
  Created wheel for gym: filename=gym-0.18.0-py3-none-any.whl size=1656445 sha25
6=971f6e92af07ec0390ca3cfd53a0e68f5687fd4fc31d0faf20fdaae340e67a4a
  Stored in directory: /Users/yuetongliu/Library/Caches/pip/wheels/d8/e7/68/a3f0
f1b5831c9321d7523f6fd4e0d3f83f2705a1cbd5daaa79
Successfully built gym
Installing collected packages: pyglet, Pillow, gym, gym-go
  Attempting uninstall: Pillow
    Found existing installation: Pillow 8.0.1
    Uninstalling Pillow-8.0.1:
      Successfully uninstalled Pillow-8.0.1
  Running setup.py develop for gym-go
Successfully installed Pillow-7.2.0 gym-0.18.0 gym-go pyglet-1.5.0
```

Rules

1. The board is empty at the onset of the game (unless players agree to place a handicap).
2. Black makes the first move, after which White and Black alternate.
3. A move consists of placing one stone of one's own color on an empty intersection on the board.
4. A player may pass their turn at any time.
5. A stone or solidly connected group of stones of one color is captured and removed from the board when all the intersections directly adjacent to it are occupied by the enemy. (Capture of the enemy takes precedence over self-capture.)
6. No stone may be played so as to recreate a former board position.

7. Two consecutive passes end the game.
8. A player's area consists of all the points the player has either occupied or surrounded.
9. The player with more area wins.

Visulization

In [48]:
```
import gym

go_env = gym.make('gym_go:go-v0', size=3, komi=0, reward_method='heuristic')

first_action = (0,1)
second_action = (1,0)
third_action = (1,2)
state, reward, done, info = go_env.step(first_action)
state, reward, done, info = go_env.step(second_action)
state, reward, done, info = go_env.step(third_action)
go_env.render('terminal')
```

```
    0   1   2
  -------------
0 | . | B | . |
  -------------
1 | W | . | B |
  -------------
2 | . | . | . |
  -------------
         Turn: W, Last Turn Passed: False, Game Over: 0
         Black Area: 3.0, White Area: 1.0
```

Reward

The reward methods are in black's perspective

Real:

-1 - White won

0 - Game is tied

1 - Black won

Heuristic:

If the game is ongoing, the reward is black area - white area. If black won, the reward is BOARD_SIZE^2. If white won, the reward is -BOARD_SIZE^2. If tied, the reward is 0.

In [46]:
```
reward
```

Out[46]:
```
1.0
```

State

The state object that is returned by the reset and step functions of the environment is a 6 x BOARD_SIZE x BOARD_SIZE numpy array. All values in the array are either 0 or 1

First channel: represent the black pieces.

Second channel: represent the white pieces.

Third channel: Indicator layer for whose turn it is.

Fourth channel: Invalid moves (including ko-protection) for the next action.

Fifth channel: Indicator layer for whether the previous move was a pass.

Sixth channel: Indicator layer for whether the game is over.

In [34]:
```
state
```

Out[34]:
```
array([[[0., 1., 0.],
        [0., 0., 1.],
        [0., 0., 0.]],

       [[0., 0., 0.],
        [1., 0., 0.],
        [0., 0., 0.]],

       [[1., 1., 1.],
        [1., 1., 1.],
        [1., 1., 1.]],

       [[0., 1., 1.],
        [1., 0., 1.],
        [0., 0., 0.]],

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]],

       [[0., 0., 0.],
        [0., 0., 0.],
        [0., 0., 0.]]])
```

Action

The step function takes in the action to execute and can be in the following forms:

a tuple/list of 2 integers representing the row and column or None for passing

In [53]:
```
action = state[3]
```

Out[53]:
```
array([[0., 1., 1.],
       [1., 0., 1.],
       [0., 0., 0.]])
```

TODO: define value funtion for state and action