GitHub: dtdannen/dcss-ai-wrapper

dustin.dannenhauer@parallaxresearch.org

# DCSS-AI-Wrapper

## An API for **D**ungeon **C**rawl **S**tone **S**oup providing both Vector and Symbolic State Representations

*Dustin Dannenhauer[1], Zohreh A. Dannenhauer[2], Jonathan Decker[3], Adam Amos-Binks[4], Michael W. Floyd[2], David W. Aha[3]*

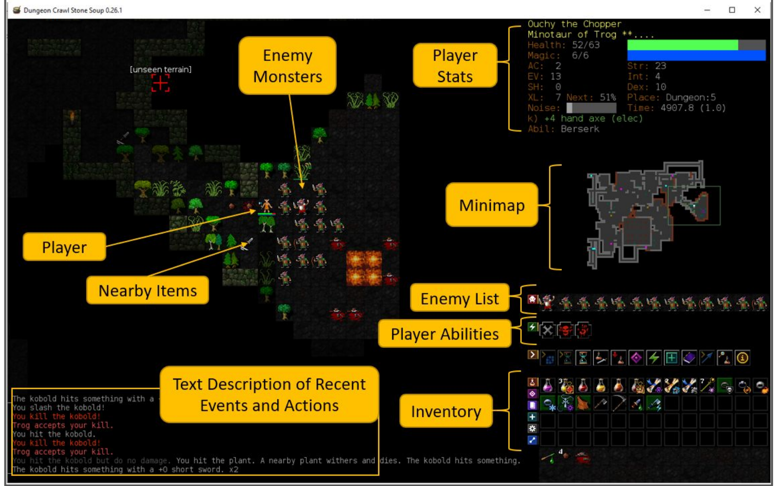
[1]Parallax Advanced Research [2]Knexus Research Corporation [3]Naval Research Laboratory [4]Applied Research Associates

## About DCSS

dcss-ai-wrapper is the first AI-friendly API for Dungeon Crawl Stone soup: a single-player, free, and open-source rogue-like video game with a variety of features that make it a challenge for artificial intelligence (AI) research:

- 2D gridworld
- Procedurally Generated
- Partially observable
- Observation actions
- Dynamic
- Stochastic
- “Wide” domain with:
  - ❏ 100’s of actions & spells
  - ❏ 650+ monster types
  - ❏ 13K+ starting characters
  - ❏ 31 skills
  - ❏ 100’s of unique items

To win requires visiting ~70k tiles



```
Ouchy the Chopper
Minotaur of Trog **....
Health: 52/63
Magic:  6/6
AC:  2               Str: 23
EV: 13               Int: 4
SH:  0               Dex: 10
XL:  7 Next: 51%     Place: Dungeon:5
Noise:               Time: 4907.8 (1.0)
k) +4 hand axe (elec)
Abil: Berserk
```

```
The kobold hits something with a
You slash the kobold!
You kill the kobold!
Trog accepts your kill.
You hit the kobold.
You kill the kobold!
Trog accepts your kill.
You hit the kobold but do no damage. You hit the plant. A nearby plant withers and dies. The kobold hits something.
The kobold hits something with a +0 short sword. x2
```

## Vector-based API

*includes*

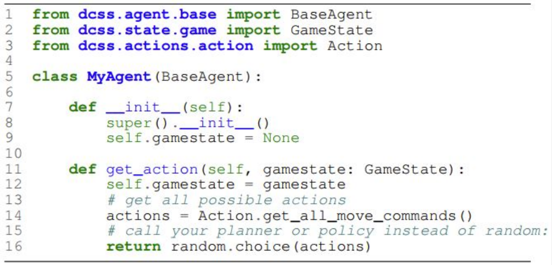
get_player_stats_vector()
get_player_inventory_vector()
get_player_spells_vector()
get_player_abilities_vector()
get_player_skills_vector()
get_LOS_map_vector()
get_level_map_vector()
get_all_map_vector()

## PDDL-based API

*includes*

get_player_stats_pddl()
get_player_inventory_pddl()
get_player_skills_pddl()
get_LOS_map_pddl()
get_level_map_pddl()
get_all_map_pddl()
...
get_background_pddl()

## Create Agents Easily



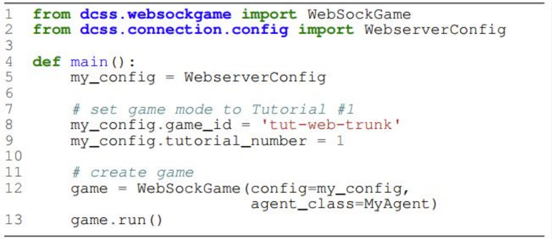
```
from dcss.agent.base import BaseAgent
from dcss.state.game import GameState
from dcss.actions.action import Action

class MyAgent(BaseAgent):

    def __init__(self):
        super().__init__()
        self.gamestate = None

    def get_action(self, gamestate: GameState):
        self.gamestate = gamestate
        # get all possible actions
        actions = Action.get_all_move_commands()
        # call your planner or policy instead of random:
        return random.choice(actions)
```

## Watch the Agent Play in the Browser



```
from dcss.websockgame import WebSockGame
from dcss.connection.config import WebserverConfig

def main():
    my_config = WebserverConfig

    # set game mode to Tutorial #1
    my_config.game_id = 'tut-web-trunk'
    my_config.tutorial_number = 1

    # create game
    game = WebSockGame(config=my_config,
                       agent_class=MyAgent)
    game.run()
```

## Paper Contributions

1) Version 0.1 of API
2) Support for both vector and PDDL-based state representations
3) First PDDL model of DCSS supporting FastDownward planner