

The background is a pixelated representation of a Super Mario Bros. level. It features a blue sky with a rainbow in the upper left, green hills, and a path made of brown bricks. In the foreground on the right, Mario is shown from the chest up, wearing his iconic red cap with a white 'M', a red shirt, and blue overalls with a yellow button. He has a wide, happy smile. The entire image has a low-resolution, pixelated aesthetic.

MarioAI

Thiago Almeida

Mario, o Jogo

- Jogo eletrônico da Nintendo, lançado em 1985
- Um dos jogos mais vendidos da história dos videogames (+40 milhões)
- Jogador controla o protagonista da série: MARIO
- Objetivo:
 - Percorrer o Reino do Cogumelo
 - Sobreviver às foras de Bowser (vilão principal)
 - Salvar a Princesa Peach

Mario, o jogo

- 8 mundos – 4 fases cada
- Ataque do Mario: pular sobre o inimigo
 - Cada inimigo pode reagir de forma diferente
 - Goomba: amassado e derrotado
 - Koopa Troopa: se esconde no casco e projeta o Mario para o alto
- Ações: direita, esquerda, pular, agachar, bomba

MarioAI – GamePlay track

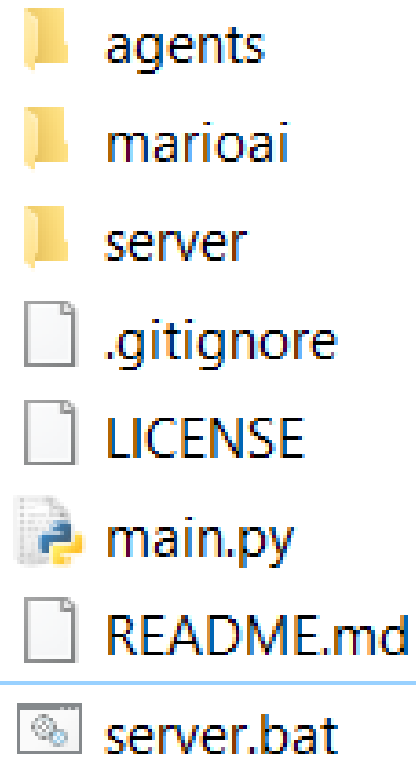
- <http://www.marioai.org/gameplay-track>
- Mario AI Championship
 - Competição de Marios inteligentes
 - GamePlay
 - Level Generation
 - Turing Test
- GamePlay – desenvolver o melhor agente como um tributo em Java para o famoso Super Mario Bros

Estrutura

- Servidor
 - Rodar o Mundo Mario
- Agente (e códigos associados)
 - Interagir com o mundo no lugar do jogador humano
- <https://github.com/renatopp/marioai>

Servidor

- Primeiro abra um terminal dentro da pasta server
- Acione o servidor com o seguinte comando:
 - `java ch.idsia.scenarios.MainRun -server on`
- Ou executar server.bat



C:\WINDOWS\system32\cmd.exe

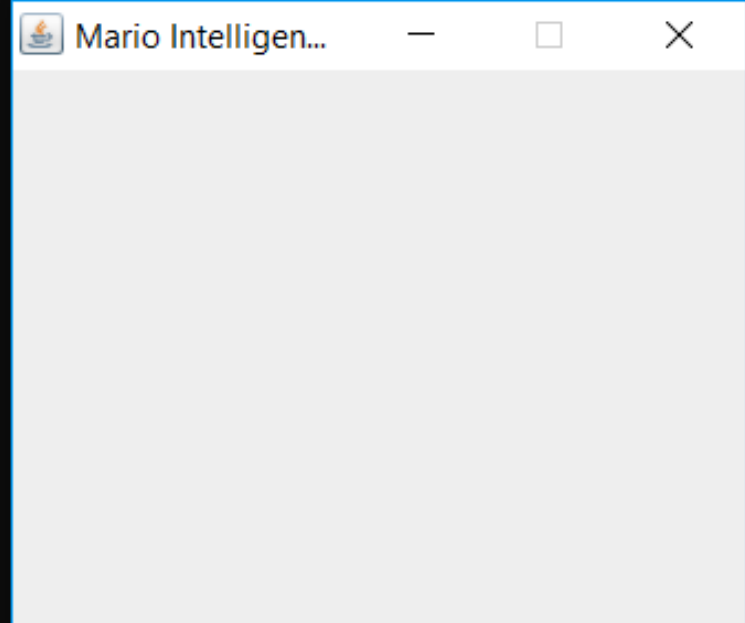
C:\Users\thiag\Downloads\W-marioai-master\marioai-master>cd server

C:\Users\thiag\Downloads\W-marioai-master\marioai-master\server>java ch.idsia.scenarios.MainRun -server on

Scoring controller SergeyKarakovskiy_JumpingAgent with starting seed 3143

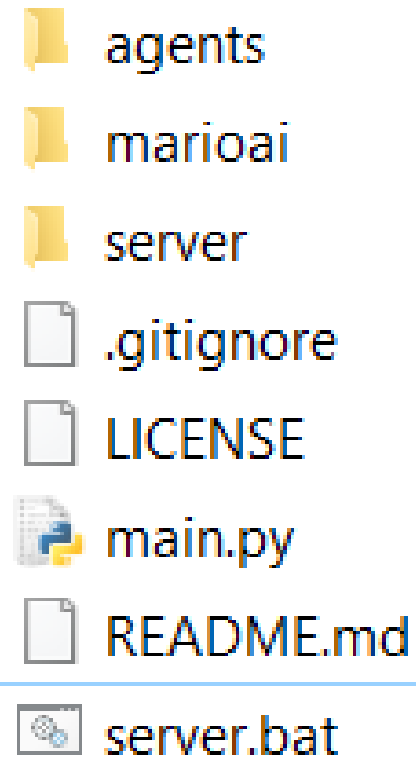
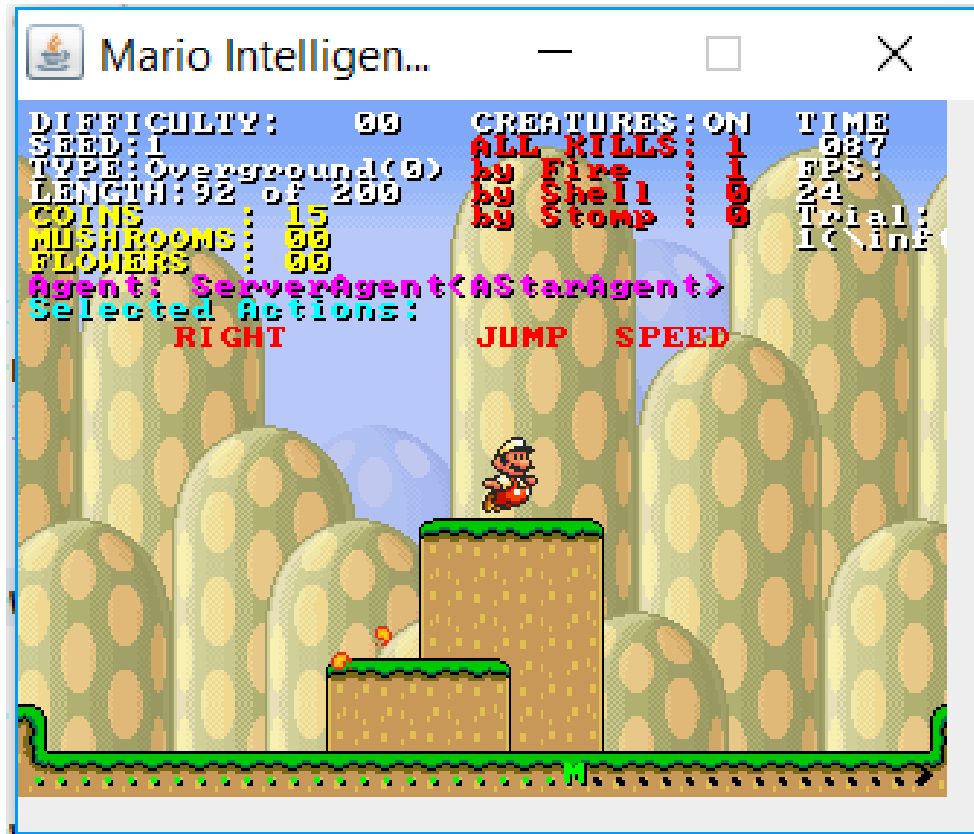
Server: Binding Server to listen port 4242

Server: Waiting for a client to connect on port 4242



Executando o Agente

- Execute o arquivo main.py
 - Utilizar Python 2



Ambiente

- `marioai/environment.py`
- Geralmente NÃO precisa de alterações
- Representa a interface do simulador MarioAI
- Atributos
 - *level_difficulty*: nível de dificuldade sendo 0 o mais fácil
 - *level_type*: tipo de nível
 - Overground: 0 (def)
 - Underground: 1
 - Castle: 2
 - Random: 3

Ambiente

- Atributos
 - *init_mario_mode*: modo inicial do Mario
 - Small: 0
 - Large: 1
 - Large with fire: 2 (def)
 - *time_limit*:
 - tempo limite que o Mario pode ficar numa fase em milisegundos
 - 100 é o default
 - *visualization*: define se o jogo vai ser visualizado no servidor ou não

Tarefa

- `marioai/task.py`
- Cuida da comunicação do agente com o mundo (environment)
- Decide como avaliar o comportamento do Mario
 - Gerencia as recompensas
- Funciona como filtro do que o agente pode ver
- Pode funcionar como filtro de como ações podem ser transmitidas para o ambiente

Tarefa

- Atributos principais
 - *env (Environment)*: instância do mundo/ambiente.
 - *reward (int)*: a recompensa atual.
 - *cum_reward (int)*: a recompense acumulada desde o início do episódio

Agente

- `marioai/agent.py`
- Classe base para um agente autônomo
- Deve ser criado um novo agente herdando dessa classe
 - `agents/novo_agente.py`

Agente

- O agente observa o mundo
- Observações são base para tomada de decisões/treinamento
- Atributos
 - *level_scene*: um numpy array 22x22 contendo todos os elementos do nível (inimigos e bloqueios).
 - *on_ground (bool)*: se o Mario está no chão ou não.
 - *can_jump (bool)*: se o Mario pode pular.
 - *mario_floats (list)*: coordenadas do Mario
 - *enemies_floats (list)*: coordenadas dos inimigos no mundo.
 - *episode_over (bool)*: se o episódio acabou.

level_scene

Valor	Significado
-11	Obstáculo fácil, pode pular através
-10	Obstáculo difícil, não dá pra pular
0	Livre, Sem obstáculos nem inimigos
1	Mario
2	Goomba (inimigo)
3	Goomba winged (inimigo)
4	Red Koopa (inimigo)
5	Red Koopa Winged (inimigo)
6	Green Koopa (inimigo)
...	...

level_scene – CADÊ O MARIO?

[illegible]

Agente - Métodos importantes

- sense
 - Recebe e analisa os sentidos do Mario
- give_rewards
 - Calcula a recompensa do Mario para ações/moedas acumuladas
 - Precisa ser implementado
- act
 - Faz o Mario executar determinada ação
 - Método mais importante
 - Onde será aplicada a inteligência do agente
 - Precisa ser implementado

Agente - Métodos importantes

- act
 - Retorna um array de 5 posições
 - Cada posição representa uma ação habilitada (1) ou desabilitada (0)
 - [voltar, em frente, agachar, pular, correr/bomba]
- act – ir para a direita
 - [0,1,0,0,0]
- act – pular para trás
 - [1,0,0,1,0]

Agente Random

```
class RandomAgent(marioai.Agent):  
    def act(self):  
        return [0, 1, 0, random.randint(0, 1), random.randint(0, 1)]
```

Agente – Árvores de Decisão

- Gerar dados de treinamento
 - Dados sensoriais do agente: mapa do ambiente, etc
- Classificar esses mapas: ação relacionada
 - Árvore ambiente perigoso/não-perigoso
 - Árvore de ação: ambiente perigoso
 - Árvore de ação ambiente não-perigoso
- Treinar o algoritmo
- Implementar o método *act*

Main

```
def main():
    agent = agents.DecisionTreeAgent()
    task = marioai.Task()
    exp = marioai.Experiment(task, agent)

    exp.visualization = True
    exp.max_fps = 20
    task.env.level_type = 0
    task.env.level_difficulty = 1
    task.env.init_mario_mode = 2
    task.env.time_limit = 100

    random.seed(20)

    #fase random - 1
    task.env.level_seed = random.randint(0, 500)
    print "Level: " + str(task.env.level_seed)
    exp.doEpisodes(1)
```

Referências

- <https://github.com/renatopp/marioai>
- <https://github.com/weslanra/marioai/>
- <http://www.marioai.org/gameplay-track>

thiagosa@dcomp.ufs.br