DOCUMENTACIÓN

Minecraft Agent Framework

Autor: Guillem Casares Valencia

Índice

Introducción

Minecraft Agent Framework es un framework para crear, gestionar y ejecutar agentes en un servidor de Minecraft. Los agentes pueden interactuar con el mundo de Minecraft, realizando acciones como enviar y recibir mensajes en el chat, teletransportar al jugador, obtener su posición, colocar bloques y destruirlos. Además, el framework proporciona un conjunto de comandos que los jugadores pueden ejecutar directamente desde el chat del juego, permitiéndoles interactuar con el framework de manera clara y sencilla, por ejemplo, para obtener información y ejecutar los agentes desarrollados por el programador.

Características principales

1. **Creación de agentes**: Ofrece una interfaz sencilla para desarrollar agentes que interactúen con el mundo de Minecraft.
2. **Interacción con el mundo de Minecraft**: Los agentes pueden enviar y recibir mensajes en el chat, teletransportar al jugador, obtener su posición, colocar bloques y destruirlos**.** Para esto se pueden usar métodos ya implementados en el framework, de esta manera no es necesario que el programador aprenda a utilizar la librería mcpi para poder comunicarse con el servidor.
3. **Sistema de comandos**: Se proporciona un conjunto de comandos para que el jugado pueda interactuar con el framework, por ejemplo, para ejecutar agentes.
4. **Extensibilidad**: Se pueden añadir nuevos métodos a los agentes y nuevos comandos al framework sin necesidad de modificar el resto del código base.
5. **Métodos ejecutables**: Se puede decorar un método de un agente con @executable para que este pueda ser ejecutado mediante comandos desde el chat de Minecraft.

Estructura del framework

Decoradores:

- **@executable:** Se usa para indicar que un método de un agente es ejecutable mediante comandos desde el chat de Minecraft.
- **@command (cmd:** str**):** Se usa en la clase MinecraftFramework para asociar uno de sus métodos al comando pasado como argumento al

decorador (cmd). De manera que cuando desde el chat del juego se ejecute ese comando, se ejecutará el método.

Clases principales:

- **MinecraftAgent:** Clase abstracta de la que deben heredar todos los agentes. Proporciona métodos para que estos puedan interactuar de manera sencilla con el mundo de Minecraft. Define un template method para la ejecución de agentes, con un flujo de ejecución compuesto por tres partes: inicio, ejecución principal y finalización. Los agentes pueden personalizar este flujo sobrescribiendo cada una de estas tres partes. En concreto, hay que sobrescribir como mínimo el método abstracto main_execute, que se corresponde a la ejecución principal del agente.

- **MinecraftFramework:** Clase que es el núcleo del framework, se encarga de gestionar y ejecutar los agentes creados, además de procesar los comandos escritos por el jugador y ejecutar los métodos correspondientes.

Método auxiliar para la ejecución de métodos:

- **method_execution (method, params, args):** Se encarga de ejecutar el método (method) según sus parámetros (params), pasándole los argumentos necesarios (args). Este método sirve para poder ejecutar cualquier método asociado a un comando o cualquier método de un agente decorado con @executable, teniendo en cuenta sus parámetros y los argumentos que se le pasan. De esta manera se evita tener que repetir código.

Clase MinecraftAgent

Constructor

MinecraftAgent (**name**: str, **active**: bool, **info**: str, **mc**: Minecraft)

Parámetros:

- **name:** Nombre del agente, este atributo sirve para identificar al agente cuando este se añade al framework, por lo que no podrá añadirse más de un agente con el mismo nombre.

- **active:** Estado inicial del agente, activo (True) o inactivo (False). Solo se podrá ejecutar si está activo.

- **info:** Descripción o información del agente. Por ejemplo, que hace y que parámetros hay que pasarle para poder ejecutarlo.
- **mc:** Instancia de Minecraft para interactuar con el servidor.

Dentro del constructor se crea y se inicializa la lista **executable_methods**, que contiene todos los métodos de la instancia que hayan sido decorados con @executable.

Métodos

Métodos para interactuar con el mundo de Minecraft:

- **send_message (message: str):** Envía un mensaje al chat.
- **receive_message () -> str:** Lee y devuelve el último mensaje del chat o una cadena vacía si no hay ningún mensaje.
- **tp_player (x, y, z):** Teletransporta al jugador a la posición (x, y, z).
- **get_player_pos ():** Devuelve la posición del jugador, contenida en un vector con las coordenadas x, y, z.
- **place_block (x, y, z, block_type, data=0):** Coloca un bloque del tipo block_type en la posición (x, y, z). El parámetro data indica si se trata de un bloque normal (data=0) o si tiene alguna propiedad adicional (data != 0).
- **destroy_block (x, y, z):** Destruye el bloque de la posición (x, y, z).

Método para mostrar los métodos de la clase que se pueden ejecutar mediante comandos desde el chat de Minecraft:

- **show_methods ():** Muestra en el chat los métodos de la lista executable_methods y sus parámetros.

Métodos para la ejecución de un agente:

- **execute (*args):** Template method para ejecutar un agente. Dentro de él se llama a los métodos ini_execute, main_execute y end_execute, en este orden.
- **ini_execute ():** Se ejecuta al inicio del agente.
- **main_execute (*args):** Implementación principal del agente (debe ser sobrescrita por las clases hija).
- **end_execute ():** Se ejecuta al finalizar el agente.

Clase MinecraftFramework

Constructor

MinecraftFramework (**mc**: Minecraft)

Parámetros:

- **mc:** Instancia de Minecraft para interactuar con el servidor.

Dentro del constructor se crea y se inicializa el diccionario **commands**, cuyas claves son los comandos disponibles en el framework y los valores son tuplas que contienen el método asociado al comando y sus parámetros.

También se crea una lista vacía, **agents**, que contendrá los agentes que se añadan al framework.

Métodos

- **__print_info (message):** Escribe un mensaje informativo en la consola con la fecha y hora actuales y el prefijo “[Agent Framework]: ”.
- **write_chat (message):** Escribe en el chat el string message con el prefijo “[Agent Framework]: ”.
- **read_chat () -> str:** Lee y devuelve el último mensaje del chat o una cadena vacía si no hay ningún mensaje.
- **search_agent (agent_name: str) -> MinecraftAgent:** Busca un agente por su nombre y lo devuelve. Si no lo encuentra devuelve None.
- **add_agent (agent: MinecraftAgent):** Añade un agente al framework solo si no hay ningún otro con el mismo nombre.
- **remove_agent (agent_name: str):** Elimina un agente por su nombre, si existe.
- **run ():** Ejecuta el bucle principal del framework para procesar comandos del chat.

Comandos

1. **af: help**: Ejecuta el método help(). Muestra los comandos disponibles y sus argumentos.
2. **af: shagents**: Ejecuta el método show_agents(). Muestra en el chat los nombres de todos los agentes y su estado actual.
3. **af: sayhi**: Ejecuta el método say_hi(). Los agentes activos saludan en el chat. Es útil para comprobar fácilmente des del chat que agentes están activos.

4. **af: chstate <agent_name>**: Ejecuta el método change_agent_state (agent_name: str). Cambia el estado de un agente de activo a inactivo y viceversa.
5. **af: shmethods <agent_name>**: Ejecuta el método show_methods(agent_name: str). Muestra los métodos ejecutables de un agente.
6. **af: shinfo <agent_name>**: Ejecuta el método show_info(agent_name: str). Muestra la información de un agente.
7. **af: exagent <agent_name> [args...]**: Ejecuta el método execute_agent (agent_name: str, *args). Ejecuta un agente (método execute) si existe y está activo, pasándole los argumentos contenidos en args.
8. **af: exmethod <agent_name> <method_name> [args...]**: Ejecuta el método execute_method(agent_name: str, method_name: str, *args). Ejecuta un método llamado method_name de un agente, si este existe y está activo. El método tiene que estar en la lista executable_methods del agente, es decir, tiene que haber sido decorado con @executable. A este método se le pasan los argumentos contenidos en args.

Creación de un agente

Para crear un agente, hereda de la clase MinecraftAgent y sobrescribe el método main_execute. También puedes definir métodos adicionales y decorarlos con @executable para que puedan ser ejecutados desde el chat de Minecraft. Para interactuar con el mundo de Minecraft hay que usar los métodos definidos en la clase MinecraftAgent.

Hay que tener en cuenta que los métodos pensados para ser ejecutados mediante comandos a través del chat de Minecraft, como el método main_execute o cualquier método anotado con @executable, reciben sus argumentos como objetos de tipo str (string). Por lo que el programador deberá convertir los argumentos al tipo adecuado, lanzando una excepción en caso de error.

Por último, hay que mencionar que en estos métodos no se admite el parámetro **kwargs, para que acepten un número variable de argumentos hay que usar *args, como en el caso de main_execute. En este caso habrá que comprobar que el número de elementos en args sea igual al número de argumentos necesarios y lanzar una excepción en caso contrario. Ejemplo:

```python
from framework.MinecraftAgentFramework import MinecraftAgent, executable

class MyAgent(MinecraftAgent):
    def __init__(self, name, active, mc):
        info = "Agente personalizado"
        super().__init__(name, active, info, mc)

    def main_execute(self, *args):
        self.send_message("Ejecutando agente personalizado")

    @executable
    def custom_method(self):
        self.send_message("Método personalizado ejecutado")
```

Añadir agentes al framework y ejecutarlo

Usa la libreria mcpi para crear una instancia de la clase Minecraft, que es la que permite la comunicación con el servidor. Luego crea una instancia de MinecraftFramework, crea instancias de tus agentes, añádelos al framework y ejecútalo. Ejemplo:

```python
from framework.MinecraftAgentFramework import MinecraftFramework
from mcpi.minecraft import Minecraft
from agents.MyAgent import MyAgent

# Crear una instancia de Minecraft
mc = Minecraft.create()

# Crear una instancia del framework
framework = MinecraftFramework(mc)

# Crear y añadir agentes
my_agent = MyAgent(name="MyAgent", active=True, mc=mc)
framework.add_agent(my_agent)

# Ejecutar el framework
framework.run()
```

Ejecutar comandos

Los comandos empiezan por el prefijo "**af:**", seguido del comando y de los argumentos necesarios, todo esto separado por espacios:

af: comando [arg1] [arg2] ... [argN]

Usa el comando "**af: help**" para ver todos los comandos disponibles y los argumentos que reciben.