Options for running your Agents

Run agent locally

Sometimes you'll want to run an agent on your own hardware or infrastructure; luckily this is very easy to do on any system that support Python 3.10.

This system is pretty simple, as to get you started as quickly as possible. We're going to run this agent on any device you'd like, in this scenario we're running on a VM but you could run this on your laptop, raspberry pi or tweak for Agentverse. On startup our script will register our agent to theAlmanac ↗ , and then our agent will be available to communicate with other agents.

The agent

agent.py from uagents . setup import fund_agent_if_low from uagents import Agent , Context , Protocol , Model import random from pydantic import Field from ai_engine import UAgentResponse , UAgentResponseType import sys

agent

Agent (name = "dungeonsanddragonsdiceroll" , port = 6145 , seed = "RANDOM STRINGS" , endpoint = ["http://YOUR_IP:6145/submit"],)

fund_agent_if_low (agent.wallet. address ())

@agent . on_event ("startup") async

def

hi (ctx : Context): ctx . logger . info (agent.address)

class

Request (Model): dice_sides :

int

=

Field (description = "How many sides does your dice need?")

dice_roll_protocol

Protocol ("DungeonsAndDragonsDiceRoll")

@dice_roll_protocol . on_message (model = Request, replies = {UAgentResponse}) async

def

roll_dice (ctx : Context ,

sender :

str ,

msg : Request): result =

str (random. randint (1 , msg.dice_sides)) message =

f "Dice roll result: { result } " await ctx . send (sender, UAgentResponse (message = message, type = UAgentResponseType.FINAL))

agent . include (dice_roll_protocol, publish_manifest = True)

agent . run () A few things to note; you'll need to be running this agent on infrastructure that allows you to open a port, in our example we run on port6145 .

The agent is initialised with an endpoint, and a port - this is so that we can receive messages, and other agents know where to send them. We callfund_agent_if_low to get some funds, if we need them. And we define our protocol, which is just an int

as seen in theRequest object.

Ouron_message() doesn't do much other than return a number between 1 and the defineddice_sides from the message inclusive. However, the response type is ofUAgentResponse which is essential to communicate with DeltaV.

.run() initialises the agent.

Finally, we run our agent as follows:python agent.py

Expected output :

INFO: [dungeonsanddragonsdiceroll]: Manifest published successfully: DungeonsAndDragonsDiceRoll INFO: [dungeonsanddragonsdiceroll]: Registering on almanac contract... INFO: [dungeonsanddragonsdiceroll]: Registering on almanac contract...complete INFO: [dungeonsanddragonsdiceroll]: agent1qvh76795enwgnzkrjpedlnqxwv83d8wxnkkcszs9z46zc3qpfs3yvzc5kuw INFO: [dungeonsanddragonsdiceroll]: Starting server on http://0.0.0.0:6145 (Press CTRL+C to quit)

Run agents with Agentverse mailbox

Through theAgentverse you can set up mailboxes for your local agents, allowing them to communicate with other agents registered on the Fetch Network without the need to be constantly online and without requiring your constant presence to operate.

Local agent setup

Let's start by creating a local agent namedalice with ahandle_message() function using an@agent.on_message() decorator to handle messages received by other agents and matching theMessage class:

from uagents import Agent , Context , Model

class

Message (Model): message :

str

First generate a secure seed phrase (e.g. https://pypi.org/project/mnemonic/)

SEED_PHRASE

"put_your_seed_phrase_here"

Copy the address shown below

print (f "Your agent's address is: { Agent (seed = SEED_PHRASE).address } ")

Then go to https://agentverse.ai, register your agent in the Mailroom

and copy the agent's mailbox key

AGENT_MAILBOX_KEY

"put_your_AGENT_MAILBOX_KEY_here"

Now your agent is ready to join the agentverse!

agent

Agent (name = "alice" , seed = SEED_PHRASE, mailbox = f " { AGENT_MAILBOX_KEY } @https://agentverse.ai" ,)

@agent . on_message (model = Message, replies = {Message}) async

def

handle_message (ctx : Context ,

sender :

str ,

msg : Message): ctx . logger . info (f "Received message from { sender } : { msg.message } ")

send the response

ctx . logger . info ("Sending message to bob") await ctx . send (sender, Message (message = "hello there bob"))

if

name

==

"**main**" : agent . run () You can easily create a Mailbox by first retrieving your local agent address and head over to theAgentverse: My Agents tab. Here, click onLocal Agents and click onConnect Local Agent . You will need to provide the address of the local agent you wish to retrieve and wait for confirmation.

You will then need to provide aname for the agent. Once you do so and confirm, you will see aMailbox API Key showing up. Copy and paste it within your local agent code above by filling up theAGENT_MAILBOX_KEY field inline.

You can then restart your agent.

ℹ You can find additional information on the Agentverse Mailbox feature here ↗ . This resource shows both the local and Agentverse agents setups to allow for a two-way communication line between these two agents.

Was this page helpful?

Send tokens with Agents Agents Mailboxes