SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
VARAŽDIN

Sara Sušac

VISEAGENTNI SUSTAV ZA MJERENJE MOTORIČKIH SPOSOBNOSTI PUTEM PISANJA NA RAČUNALU

PROJEKT

VIEAGENTNI SUSTAVI

Varaždin, 2024.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Sara Sušac

Matični broj: 0016143183

Studij: Baze podataka i baze znanja

VISEAGENTNI SUSTAV ZA MJERENJE MOTORIČKIH SPOSOBNOSTI PUTEM PISANJA NA RAČUNALU

PROJEKT

Mentor:

dr. sc. Bogdan Okresa Đurić

Varaždin, siječan 2024.

Sara Sušac

Izjava o izvornosti

Izjavljujem da je ovaj projekt izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autorica potvrdila prihvaćanjem odredbi u sustavu FOI Radovi

Sažetak

U ovome projektu izraditi ćemo igru koja će služiti za poboljšanje motoričkih sposobnosti tako da se mjeri brzina pisanja korisnika u konzoli. Glavna struktura programa je razmijena infomacija između dva agenta.

Ključne riječi: Python, Spade, Agenti, Openfire

Sadržaj

1. Uvod

Projekt predstavlja interaktivnu računalnu igru dizajniranu s ciljem poticanja i mjerenja motoričkih sposobnosti kod korisnika. Kroz inovativan pristup, korisnici će imati priliku razvijati svoje motoričke vještine kroz dinamične izazove i igre, stvarajući istovremeno zabavno i edukativno iskustvo.

U okviru projekta, agenti predstavljaju autonomne računalne entitete odgovorne za različite zadatke. Agenti su programski entiteti koji komuniciraju, surađuju i izvršavaju specifične funkcije unutar interaktivne računalne igre. Primarni cilj agenata je omogućiti dinamičnost, praćenje korisničkih postignuća te ostvariti kvalitetno iskustvo korisnika kroz mjerenje motoričkih sposobnosti i prilagodljivost igre tijekom različitih razina.

2. Opis korištenih tehnologija

U ovome poglavlju ukratko ćemo objasniti programski jezik Python, okruženje za razvoj inteligentih agenata Spade te Openfire, kolaboracijski server otvorenog koda baziran na XMPP standardu.

2.1. Python

Python je visokonivou programski jezik koji je popularan zbog svoje jednostavnosti i svestranosti. Omogućuje brzo pisanje i čitanje koda, što ga čini izvrsnim izborom za različite vrste projekata. Python ima bogat ekosustav biblioteka i okvira koji olakšavaju razvoj raznovrsnih aplikacija, uključujući web aplikacije, analizu podataka, strojno učenje, automatizaciju, i mnoge druge. [1]

Python je korišten kao glavni programski jezik za implementaciju vašeg projekta. Njegova jednostavna sintaksa čini ga idealnim alatom za razvoj aplikacije koja uključuje agente koji komuniciraju i surađuju u stvarnom vremenu. Python također olakšava integraciju s različitim tehnologijama i alatima koji su korišteni u projektu, poput Spade i Openfire.

2.2. Spade

"Spade" biblioteka za Python, koristi se za razvoj multiagentnih sustava. "Spade" (Simple Platform for Agent-based Development and Experimentation) pruža okruženje za razvoj, testiranje i implementaciju agenata. [2]

U našem projektu, Spade je korišten za stvaranje i upravljanje agentima koji sudjeluju u interaktivnoj konzolnoj igri.

2.3. Openfire

Openfire je platforma za otvorenu komunikaciju koja implementira XMPP (Extensible Messaging and Presence Protocol) standard. XMPP je protokol za razmjenu poruka i prisutnosti koji se često koristi za implementaciju sustava za trenutnu komunikaciju i kolaboraciju. Openfire pruža funkcionalnosti za postavljanje i upravljanje vlastitim XMPP serverom, omogućujući organizacijama da uspostave svoj siguran sustav za razmjenu poruka.[3]

U ovom projektu Openfire nam je služio kao server za razmijenu poruka između dva agenta kako bi omogućio njihovu koordinaciju i komunikaciju u realnom vremenu.

3. Implementacija višeagentnog sustava za mjerenje motoričkih sposobnosti

U ovome poglavlju prikazati ćemo kako smo unutar kojeg file-a isprogramirali ovu igru koristeći se gore navedenim tehnologijama. Prikazati ćemo kod unutar svakog python file-a i objasniti ga. Koristili smo 4 python fila: agent1GameSetup.py, agent2Timer.py, common.py, gamemain.py.

3.1. Agent1GameSetup.py

Isječak kôda 1: Agent1GameSetup.py

```
import spade
from common import start_new_level
from spade.behaviour import OneShotBehaviour
from spade.message import Message


class GameSetupAgent(spade.agent.Agent):
    def __init__(self, jid, password):
        super().__init__(jid, password)

    class GameSetupBehaviour(OneShotBehaviour):
        async def run(self):
            while True:
                level = 1
                symbol, duration = start_new_level(level)
                while True:
                    print(f"Level {level}: Pripremite se, slovo koje trebate
                     pritisnuti glasi: {symbol}")
                    await self.send_symbol_to_agent2(symbol, duration=duration)

                    message = await self.receive(timeout=100)
                    if message:
                        if message.metadata.get("ontology") == "result":
                            elapsed_time = float(message.body.split(" ")[0])
                            flag = int(message.body.split(" ")[1])
                            if flag == 1:
                                print(f"Točno! Vaše vrijeme je: {elapsed_time:.2f}
                                 sekundi.")
                                level += 1
                                if level == 6:
                                    print(f"\nČestitamo!!! Pobijedili ste :)")
                                    await self.send_break_to_agent2()
                                    break
                                else:
                                    print("Prelazite na novi level.\n")

                            elif flag == 0:
```

```
                                print(f"Krivo ste odgovorili ili Vam je trebalo
                                 previše vremena. Vraćate se na prvi level.\n")
                                level = 1

                    symbol, duration = start_new_level(level)

                break
            await self.agent.stop()

        async def send_symbol_to_agent2(self, symbol, duration):
            msg = Message(to="agent2@laptop-rfc8bfqq")
            msg.set_metadata("ontology", "symbol")
            msg.body = symbol + " " + str(duration)
            await self.send(msg)

        async def send_break_to_agent2(self):
            msg = Message(to="agent2@laptop-rfc8bfqq")
            msg.set_metadata("ontology", "break")
            await self.send(msg)

    async def setup(self):
        print("Agent 1 started")
        self.add_behaviour(self.GameSetupBehaviour())
```

Unutar ovog Python file-a definira se agent (Agent 1) unutar okvira razvoja sustava s više agenata pomoću "Spade" biblioteke. Uključuje uvoz modula poput spade, common, OneShotBehaviour i Message. Agent je konfiguriran putem klase GameSetupAgent, koja nasljeđuje osnovnu klasu spade.agent.Agent te postavlja JID (Jabber ID) i lozinku agenta.

Unutar klase GameSetupAgent, definirana je unutarnja klasa GameSetupBehaviour, nasljeđujući OneShotBehaviour. Metoda run ove klase sadrži logiku igre koja se izvršava u beskonačnoj petlji, gdje agent šalje simbole agentu 2 i reagira na odgovore. Ovisno o odgovoru, agent može preći na sljedeći level ili završiti igru.

Dodatno, metode sendsymboltoagent2 i sendbreaktoagent2 koriste se za slanje određenih vrsta poruka agentu 2. Metoda setup postavlja agenta dodavanjem prethodno definiranog ponašanja.

3.2. Agent2Timer.py

Isječak kôda 2: Agent2Timer.py

```
import random

import spade
import time
from common import check_answer, measure_time
from spade.behaviour import OneShotBehaviour
from spade.message import Message

```

```
9
10 class TimerAgent(spade.agent.Agent):
11     def __init__(self, jid, password):
12         super().__init__(jid, password)
13
14     class TimerBehaviour(OneShotBehaviour):
15         async def run(self):
16             while True:
17                 message = await self.receive(timeout=100)
18
19                 if message:
20                     if message.metadata.get("ontology") == "symbol":
21                         symbol = message.body.split(" ")[0]
22                         duration = message.body.split(" ")[1]
23
24                     if message.metadata.get("ontology") == "break":
25                         break
26
27                 while True:
28                     symbols = ["A", "B", "C", "D", "E", "F", "G", "H", "I", "J"]
29                     test = random.choice(symbols)
30
31                     print(test)
32
33                     if test == symbol:
34                         start_time = time.time()
35
36                     user_input = input("Pritisnite 'y' kada vidite ispravan simbol
                        ili 'n' ako je simbol neispravan: ")
37
38                     if user_input == 'y' and check_answer(test, symbol):
39                         elapsed_time = measure_time(start_time)
40                         if elapsed_time <= float(duration):
41                             await self.send_result_to_agent1(elapsed_time, 1)  # 1
                                oznacava da je tocno odgovorio
42                             break
43
44                         elif elapsed_time > float(duration):
45                             await self.send_result_to_agent1(elapsed_time, 0)  # 0
                                oznacava da je krivo odgovorio
46                             break
47
48                     elif user_input == 'y' and not check_answer(test, symbol):
49                         await self.send_result_to_agent1(0, 0)  # 0 oznacava da je
                            krivo odgovorio
50                         break
51
52                     elif user_input == 'n' and test == symbol:
53                         await self.send_result_to_agent1(0, 0)  # 0 oznacava da je
                            krivo odgovorio
54                         break
55
56                     elif user_input == 'n':
```

```
57                          continue
58
59                      else:
60                          await self.send_result_to_agent1(0, 0)  # 0 oznacava da je
                             krivo odgovorio
61                          break
62
63              await self.agent.stop()
64
65          async def send_result_to_agent1(self, elapsed_time, flag):
66              msg = Message(to="agent1@laptop-rfc8bfqq")
67              msg.set_metadata("ontology", "result")
68              msg.body = str(elapsed_time) + " " + str(flag)
69              await self.send(msg)
70
71      async def setup(self):
72          print("Agent 2 started")
73          self.add_behaviour(self.TimerBehaviour())
```

Agent2 je konfiguriran putem klase TimerAgent, koja nasljeđuje osnovnu klasu spade.agent.Agent i postavlja JID (Jabber ID) i lozinku agenta.

Unutar klase TimerAgent, definirana je unutarnja klasa TimerBehaviour, koja nasljeđuje OneShotBehaviour. Metoda run ove klase sadrži logiku rada agenta, uključujući reakciju na primljene poruke od Agent 1. Agent generira slučajne simbole i postavlja vremensko ograničenje za odgovaranje.

Metoda sendresulttoagent1 koristi se za slanje rezultata igre Agentu 1. Šalje poruku s informacijama o vremenu proteklom tijekom igre i oznakom zastave (flag) koja označava ispravan ili neispravan odgovor.

Metoda setup koristi se za postavljanje agenta, dodajući mu prethodno definirano ponašanje (TimerBehaviour).

3.3. Common.py

Isječak kôda 3: Common.py

```
1  import time
2  import random
3
4  current_symbol = None
5  correct_answer = None
6
7
8  def start_new_level(level=1):
9      # Generiranje novog simbola za svaku razinu
10     symbols = ["A", "B", "C", "D", "E", "F", "G", "H", "I", "J"]  # Možete
        promijeniti popis simbola prema svojim potrebama
11     new_symbol = random.choice(symbols)
12     duration = 6 - level
```

```
13
14      return new_symbol, duration
15
16
17  def check_answer(user_input, current_symbol):
18      return user_input == current_symbol
19
20
21  def measure_time(start_time):
22      return time.time() - start_time
```

U datoteci common.py su definirane zajedničke funkcije i varijable koje se koriste u oba agenta, agent1GameSetup.py i agent2Timer.py, kako bi se osigurala koordinacija u njihovoj interakciji tijekom igre.

Prva funkcija, startnewlevel(level=1), ima zadaću generirati novi simbol za svaku razinu igre. Simbol se odabire nasumično iz liste simbola (A, B, C, ..., J), a trajanje simbola varira prema razini igre. Funkcija vraća novo generirani simbol i odgovarajuće trajanje simbola.

Druga funkcija, checkanswer(userinput, currentsymbol), ima ulogu usporedbe korisničkog unosa s trenutnim simbolom kako bi provjerila točnost odgovora. Vraća True ako je odgovor ispravan, odnosno False ako nije.

Treća funkcija, measuretime(starttime), koristi se za mjerenje proteklog vremena od trenutka kada se pojavio traženi simbol na kozoli do trenutka kada je koristik upisao točan odgovor u konzolu. Ova funkcionalnost je ključna za određivanje vremena odgovaranja igrača.

Globalne varijable currentsymbol i correctanswer koriste se kao međuspremnik informacija između agenata. Agent 1 postavlja currentsymbol prije slanja simbola Agentu 2, dok Agent 2 koristi istu varijablu za usporedbu tijekom igre. Ovo omogućuje sinkronizaciju informacija o trenutnom simbolu između agenata.

Ova zajednička datoteka, common.py, stvara temelje za konzistentno izvršavanje igre između agenata, pružajući im zajedničke funkcionalnosti za generiranje simbola, provjeru odgovora i mjerenje vremena.

3.4. Gamemain.py

Isječak kôda 4: Gamemain.py

```
1  import spade
2  from agent1_GameSetup import GameSetupAgent
3  from agent2_Timer import TimerAgent
4
5
6  async def main():
7      primatelj = GameSetupAgent("agent1@laptop-rfc8bfqq", "tajna")
8      posiljatelj = TimerAgent("agent2@laptop-rfc8bfqq", "tajna")
9
10     await posiljatelj.start()
```

```
11      await primatelj.start()
12
13      await spade.wait_until_finished(primatelj)
14      await posiljatelj.stop()
15      print("Gotovo")
16
17
18  if __name__ == "__main__":
19      spade.run(main())
```

Ova Python skripta služi kao izvršna datoteka koja stvara i pokreće dva agenata unutar Spade platforme. Prva klasa agenta, GameSetupAgent (Agent 1), je odgovorna za postavljanje igre, dok druga klasa agenta, TimerAgent (Agent 2), mjeri vrijeme tijekom igre.

U funkciji main(), stvaraju se instance agenata s odgovarajućim JID-ovima ("agent1@laptop-rfc8bfqq" i "agent2@laptop-rfc8bfqq") te lozinkama ("tajna"). Nakon toga, agenti se pokreću pomoću naredbi await posiljatelj.start() i await primatelj.start().

Zatim, čeka dok Agent 1 (GameSetupAgent) ne završi svoje zadatke pomoću await spade.waituntilfinished(primatelj). Nakon toga, Agent 2 (TimerAgent) se zaustavlja naredbom await posiljatelj.stop(). Konačno, ispisuje se poruka "Gotovo", označavajući završetak izvršavanja svih operacija.

Poziv spade.run(main()) pokreće glavnu asinkronu funkciju main() unutar Spade platforme. Ova izvršna datoteka koordinira interakciju između oba agenta unutar Spade platforme, čime omogućuje njihovu suradnju i izvršavanje zadataka u skladu s definiranim ponašanjem.

3.5. Način provedbe igre

Na slici 1. možemo što smo dobili kada smo ispravno prošli prvi jedan od levela igre. Dakle od korisnika se traži da ako je dani simbol jednak onom generiranom, pritisne y, a u ostalim slučajevima n. Korisnik također prelazi na idući level.

Na slici 2. možemo što smo dobili kada nismo prošli prvi level igre. Korisnik je pritisnuo slovo y i enter ali to slovo se nije podudaralo sa početnim, zadanim slovo.

Na slici 3. da korisnik nije odgovorio na vrijeme koje je zadano u ovom level te se zbog toga vraća na prvi level.

Na slici 4. korisnik je krivo odgovorio na upit te ga se zbog toga vraća na prvu razinu.

Na slici 5. korisnik je došao do kraja igre i time pobjedio.

```
Level 2: Pripremite se, slovo koje trebate pritisnuti glasi: F
B
Pritisnite 'y' kada vidite ispravan simbol ili 'n' ako je simbol neispravan: n
D
Pritisnite 'y' kada vidite ispravan simbol ili 'n' ako je simbol neispravan: n
B
Pritisnite 'y' kada vidite ispravan simbol ili 'n' ako je simbol neispravan: n
I
Pritisnite 'y' kada vidite ispravan simbol ili 'n' ako je simbol neispravan: n
I
Pritisnite 'y' kada vidite ispravan simbol ili 'n' ako je simbol neispravan: n
D
Pritisnite 'y' kada vidite ispravan simbol ili 'n' ako je simbol neispravan: n
F
Pritisnite 'y' kada vidite ispravan simbol ili 'n' ako je simbol neispravan: y
Točno! Vaše vrijeme je: 2.06 sekundi.
Prelazite na novi level.
```

Slika 1: Ispravno prođen level

```
Level 3: Pripremite se, slovo koje trebate pritisnuti glasi: C
E
Pritisnite 'y' kada vidite ispravan simbol ili 'n' ako je simbol neispravan: y
Krivo ste odgovorili ili Vam je trebalo previše vremena. Vraćate se na prvi level.

Level 1: Pripremite se, slovo koje trebate pritisnuti glasi: J
```

Slika 2: Neispravno odgovoreno

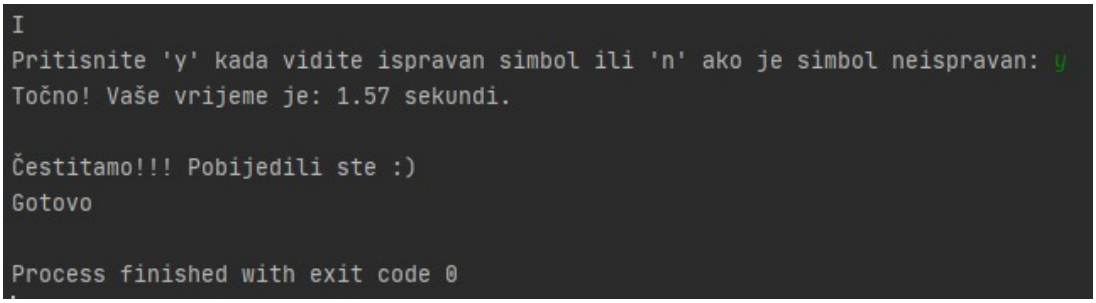
```
Level 1: Pripremite se, slovo koje trebate pritisnuti glasi: J
F
Pritisnite 'y' kada vidite ispravan simbol ili 'n' ako je simbol neispravan: n
C
Pritisnite 'y' kada vidite ispravan simbol ili 'n' ako je simbol neispravan: n
J
Pritisnite 'y' kada vidite ispravan simbol ili 'n' ako je simbol neispravan: y
Krivo ste odgovorili ili Vam je trebalo previše vremena. Vraćate se na prvi level.

Level 1: Pripremite se, slovo koje trebate pritisnuti glasi: G
```

Slika 3: Istek vremena

```
Level 1: Pripremite se, slovo koje trebate pritisnuti glasi: G
H
Pritisnite 'y' kada vidite ispravan simbol ili 'n' ako je simbol neispravan: s
Krivo ste odgovorili ili Vam je trebalo previše vremena. Vraćate se na prvi level.
```

Slika 4: Krivi odgovor na upit



Slika 5: Kaj igre

4. Zaključak

Projekt koji uključuje dva Spade agenata, GameSetupAgent i TimerAgent, ostvaruje interaktivnu igru koja je korisna za razvoj motoričkih sposobnosti i poboljšanje pamćenja. Kroz igru, korisnici rješavaju zadatke povezane s prepoznavanjem simbola, vremenskim ograničenjima i brzim motoričkim odgovorima. Ova vrsta aktivnosti potiče koordinaciju između uma i tijela, pridonoseći tako motoričkom razvoju, dok istovremeno izaziva pamćenje kroz prepoznavanje i upamćivanje simbola. Ova interaktivna igra predstavlja inovativan pristup koristeći Spade platformu za razvoj edukativnih i zabavnih igara koje imaju pozitivan utjecaj na kognitivne i motoričke vještine korisnika.

Popis literature

[1] „Welcome to Python.org.” (), adresa: https://www.python.org/.

[2] „SPADE — SPADE 3.3.0 documentation.” (), adresa: https://spade-mas.readthedocs.io/en/latest/readme.html.

[3] „Ignite Realtime: Openfire Server.” (), adresa: https://www.igniterealtime.org/projects/openfire/.

Popis slika

Popis isječaka koda