Pong Multiplayer

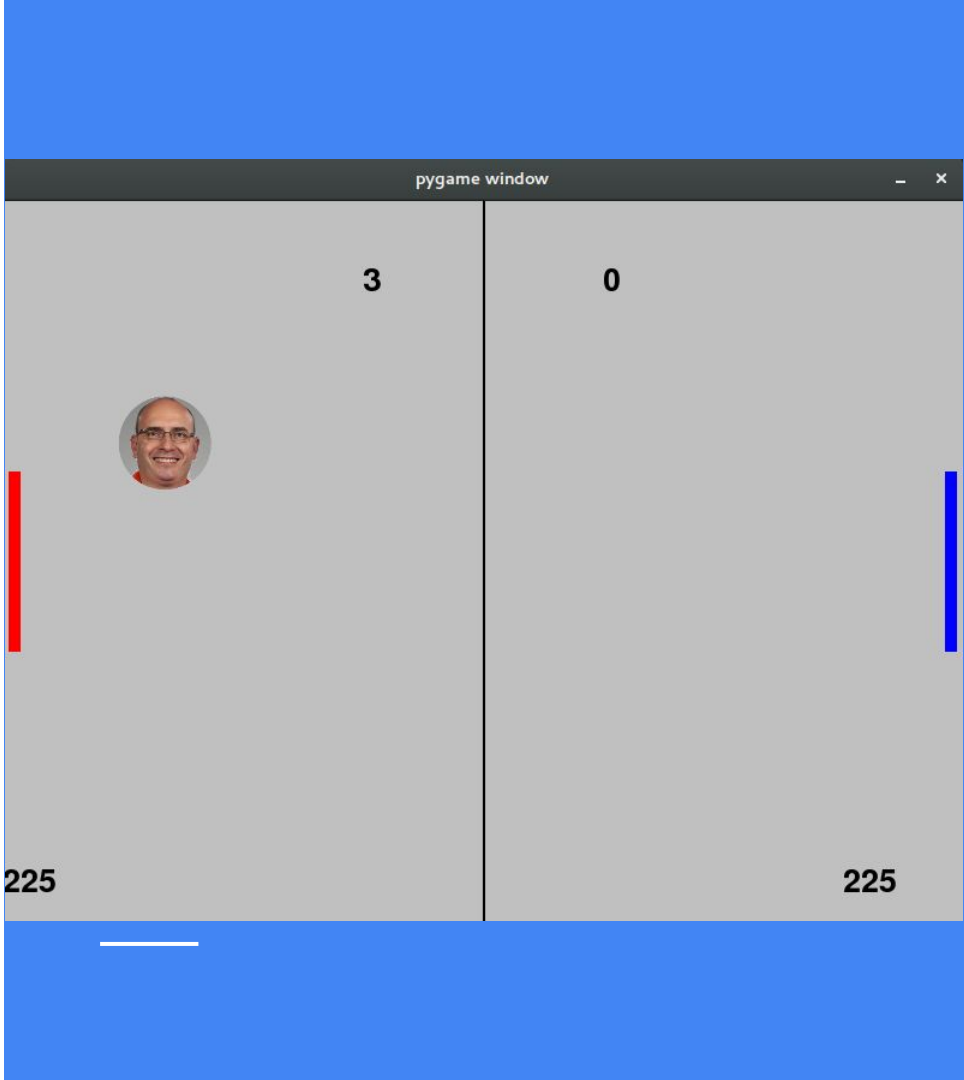
Moisés Goulart de Oliveira

Um jogo de pong multiplayer utilizando socket para comunicação

Interface

- Parte visual do jogo controlada pelo Python e desenhada com PyGame

- Controle utilizando suporte do pygame



Servidor

O servidor é responsável pela dinâmica dos objetos na tela

O servidor envia as coordenadas dos elementos para os clientes

```
cserver.start(porta)

cserver.add(int(self.player1.p),\
int(self.player1.y),int(self.player2.y), \
int(self.bola.x), int(self.bola.y), \
int(self.player1.score), int(self.player2.score), 1)
```

```
static PyObject *cserver_start(PyObject *self, PyObject *args){
    pthread_t con;
    if (!PyArg_ParseTuple(args, "i", &portno))
        return NULL;
    printf("cserver, iniciando na porta %d!\n", portno);
    pthread_create(&con, NULL, conn, NULL);
    Py_RETURN_NONE;
```

Servidor

Envia constantemente o buffer das posições

É uma tarefa periódica que envia a 60FPS, que é a frequência de atualização da tela

```
void *cliente(void *arg){
    int cid = (int)arg;
    int i, n;
    char aux[2];
    struct periodic_info info;
    make_periodic(15000, &info);
    while (1) {
            for (i = 0;i < 2; i++){
                n = send(newsockfd[cid],buffer,T_BUFF,0);
                if(n < 0){
                    printf("Erro enviando socket!\n");
                    exit(1);
                }
            }
        pthread_mutex_unlock(&mutex);
    wait_period(&info);
    }
}
```

Cliente

O Cliente é responsável por captar os comandos do teclado e enviar para o servidor

Somente recebe as coordenadas e desenha na tela

```
def controls(self):
    stop = 3
    up = 1
    down = 2
    for event in pygame.event.get():
        if event.type == KEYDOWN:
            if event.key == K_UP:
                cclient.move(up) #up
            elif event.key == K_DOWN:
                cclient.move(down) #down
        if event.type == KEYUP:
            if event.key == K_UP:
                cclient.move(stop) #stop
            elif event.key == K_DOWN:
                cclient.move(stop)
        if event.type == pygame.QUIT:
            #return True
            exit()
```

Cliente

O programa em Python envia o comando para o modulo em C que somente atualiza um buffer e envia constantemente para o servidor.

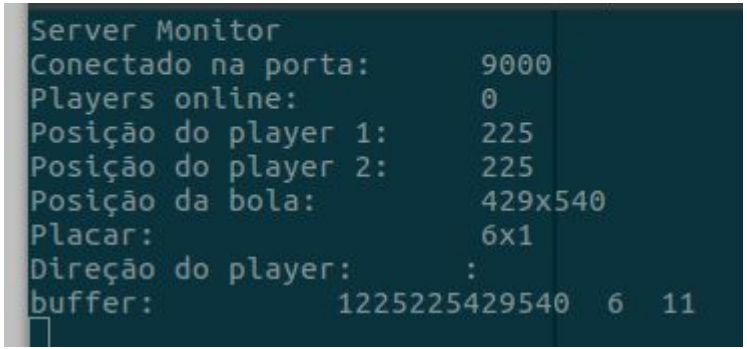
Essa tarefa é periodica

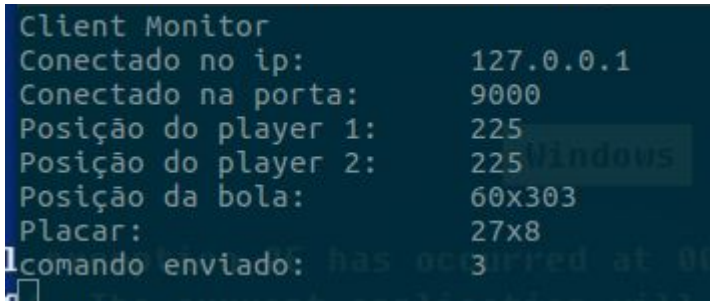
```
static PyObject *cclient_move(PyObject *self, PyObject *args){
    if (!PyArg_ParseTuple(args, "i", &dir))
        return NULL;
    Py_RETURN_NONE;
}
```

```
    do {
        bzero(direction,sizeof(direction));
        sprintf(direction, "%d",dir);
        if (send(sockfd,direction,2,0) == -1){
            printf("Erro escrevendo no socket!\n");
            exit(1);
        }
        wait_period(&info);
    }while (1);
```

Outras rotinas

Outras duas tarefas periódicas sao que responsáveis por mostrar no terminar o status do jogo.





github.com/moisesoliveira/SOTR