Prezada profª Anolan, boa noite!

Para executar nosso joguinho, a senhora precisa do node.js instalado em sua máquina com as seguintes dependencias:

express
engine.io
eureca.io

para instalar o node: sudo apt-get install node (ou node-js não sei)

para instalar os pacotes

npm install express
npm install engine.io
npm install eureca.io

para executar o server:

node server.js

para "jogar" o joguinho:

chrome localhost:8000

Eis o relatório parcial das atividades desenvolvidas por mim e pelo André Brait no trabalho prático 2.

Nós estudamos o framework Phaser, aprendemos alguns conceitos do desenvolvimento de jogos, conceitos de sprite, conceitos da física ARCADE e outras funcionalidades que serviram para o nosso proveito.

Para essa etapa do trabalho, a senhora pediu para que desenvolvesse-mos uma versão inicial de um jogo similar ao agar.io implementando mutex em regiões do mapa do jogo, para não permitir que dois jogadores ocupassem o mesmo lugar (inicialmente).

Você não pediu para que ficasse bonitinho, seria apenas interessante que ficasse assim, e conseguimos de alguma forma, implementar as funcionalidades básicas do agar.io: Carregar um mapa, adicionar jogadores no mapa e fazer esses jogadores perseguirem o ponteiro do mouse no momento quando o ponteiro é clicado.

Fizemos primeiramente o modelo para um jogador e funcionou de maneira adequada, utilizando as bibliotecas padrões do Phaser e um servidor web rodando em python.

Após isso, procuramos na internet soluções para jogos multijogadores no phaser e encontramos um exemplo dum jogo muito famoso (Tanks) e ele nos auxiliou muito para implementar nossa solução.

O jogo foi proposto pelo criador da biblioteca de nodejs eureca.io para demonstrar as funcionalidades de RPC disponíveis na biblioteca em um jogo multiplayer do phaser.
(Artigo: http://ezelia.com/2014/tutorial-creating-basic-multiplayer-game-phaser-eureca-io)

Por ser uma biblioteca de node, tivemos de mudar nosso servidor web para node.js e seguindo o exemplo proposto, implementamos a logica de RPC da biblioteca eureca.io no nosso jogo e felizmente conseguimos colocar vários jogadores no mapa ao mesmo tempo.

Porém, os mecanismos de exclusão mútua ainda não conseguimos implementar, pois estamos tentando usar as mecanicas de colisao da biblioteca Phaser, que auxiliariam e muito o nosso trabalho, porém, não estamos tendo sucesso.

Até o momento, não colocamos alimentos no mapa, somente jogadores e por isso, criamos até agora apenas uma classe para representar os jogadores (personagens do agar.io). A classe se chama Ameba e está no arquivo ascan.io.js. Ela tem várias propriedades e métodos e só quero ressaltar que já implementamos dois métodos que tendem a simular de alguma forma o funcionamento das mecanicas de velocidade e alimentação do agar.io.

Sao eles: Ameba.velocidade e Ameba.alimentar (Que ainda nao tem codigo, mas ja tem assinatura rs)

O metodo velocidade retorna uma velocidade entre 0 e 200 px/ciclo regida por uma função senoidal em funcao do tempo (depois será mudado para exponencial decrescente em funcao da massa) e a velocidade rege o scale da ameba, que aumenta e encolhe de tamnho em funcao do tempo.

O Nome provisorio do projeto é ascan.io, um trocadilho infame entre o meu sobrenome e o sufixo io presente em todos esses jogos html5 multiplayer.

Vamos continuar trabalhando no trabalho e impondo melhorias no código e vamos entregar um relatório de verdade rs.