

Relatório do Projeto – Batalha Pokémon

Fase 2 – Server and Client

Nomes:

Eduardo Delgado Coloma Bier
Fernanda de Camargo Magano
Florence Alyssa Sakuma Shibata
Shayenne da Luz Moura

Nusp:

8536148
8536044
7971705
8536235

-----CLASSES E DINÂMICA-----

- Foram criadas as classes cliente e servidor;
- É necessário o xml para atualização dos pokemons e funciona da seguinte maneira: o cliente lê seu pokemon e envia o objeto battle_state via Post para o servidor. A resposta do servidor é um objeto battle_state com dois pokémons.
- A ideia da batalha é que com base nessa resposta do servidor, o cliente escolha o ataque que achar mais adequado. Então envia para o servidor, o qual contabiliza e contra-ataca.

-----CLASSE CLIENTE-----

- Uma das responsabilidades dessa classe é escrever o xml. Para essa finalidade foi criada uma árvore e acrescentados os nós. Cria-se o objeto battle_state. Assim, ET.SubElement foi usado para isso. O '.text' foi necessário para a edição do xml.
- Desse modo, nome, tipo, ataques, defesa, PP, entre outros são alterados durante a batalha.
- No final desse método de escrita presente na classe cliente, foi utilizado o método tostring, para a conversão da árvore numa string.
- Outro método importante foi o responsável por iniciar a batalha e fazer a comunicação ao servidor por meio dos requests.

-----CLASSE SERVIDOR-----

- Lê o pokemon que já está no xml (o do cliente) e, além disso, atualiza com o seu próprio pokemon;
- É no servidor que a batalha é criada.
- Utiliza o método POST, além de importar o Flask.

-----DIFICULDADES ENCONTRADAS-----

- Inicialmente não estava muito claro o que deveria ser feito com o arquivo de extensão xsd. Tinha sido interpretado que esse seria o xml com o qual deveria-se manipular e extrair as informações necessárias. Depois de muitas pesquisas foi

compreendido qual era o intuito de um Schema e que deveria-se, a partir deste, gerar o xml;

- Outra dificuldade foi na busca de ferramentas de manipulação do xml compatível com Python 3. Foi decidido que a melhor opção seria utilizar o 'xml.etree.ElementTree as ET'.

-----TESTES-----

- Foram testadas as duas importantes classes: 'server' e 'cliente' em suas funções de leitura e escrita do xml.

-----MODO DE USO-----

- Para rodar o programa, basta fazer python3 main.py <argumento> <PikachuSolo.txt>
- No lugar de argumento use a opção: -c ou -C para o cliente e -s ou -S para o servidor;
- Lembre-se de que o flask e requests devem estar instalados.