ISEL / ADEETC

Licenciatura em Engenharia Informática e Multimédia

Produção de Conteúdos Multimédia

Aula de Laboratório

de

Produção de Conteúdos Multimédia

Jogo de Cartas - Blackjack

Rui Jesus e João Beleza

Introdução

Este trabalho visa a introdução e a familiarização com a linguagem de programação JavaScript. O objetivo é o desenvolvimento do jogo de cartas Blackjack. Cada aluno/grupo tem de implementar os métodos/funções que estão por fazer no código fornecido pelo docente. Os alunos têm liberdade para fazerem alterações desde que não se afastem muito da estrutura.
Em baixo é apresentado um *link* para o *site* do w3schools que deve ser consultado durante o desenvolvimento do jogo.
http://www.w3schools.com/

Objetivos

O projeto fornecido pelo docente 3 ficheiros: "blackjack_oop.html", "blackjack_manager.js" e "blackjack_object.js". O primeiro implementa a interface com o utilizador, o segundo faz a interface entre a interface utilizador e o objeto "blackjack" e o terceiro contém o código do objeto.

No final da primeira aula o aluno terá de ter, pelo menos, o objeto "blackjack" implementado, isto é, o aluno deverá fazer o código dos métodos do objeto. No final da segunda aula o aluno deverá ter a aplicação implementada, isto é, as funções do ficheiro "blackjack_manager.js". O **código completo do jogo** tem de ser entregue até ao dia **13 de Novembro de 2016** utilizando a plataforma Moodle.

Trabalho Laboratorial

Objeto "blackjack"

1. Unzip o ficheiro "blackjack.zip" e crie um novo projeto no IntelliJ IDEA com os ficheiros fornecidos. Faça "NEW->Projeto from Existing Sources".

2. Implemente os seguintes métodos do objeto "blackjack":
   a. `novo_baralho()`

      Este método retorna um *array* com as 52 cartas representadas por números de 1 a 13 para cada naipe. Os números 11, 12 e 13 representam as figuras (Rei, Valete e Dama).

b. `baralha()`

Este método baralha o *array* de cartas construído no método anterior e retorna um novo *array* baralhado.

Deve criar um *array* de índices de 1 a 52 (*for*). De seguida deve fazer um outro *for* que em cada ciclo (52) faça o sorteio (Math.random) de um índice. Este índice é usado para ir buscar uma carta ao baralho. Esta carta é inserida num fim de outro *array* com as cartas baralhadas. Não esquecer de remover o índice sorteado do *array* de índice.

c. `valor()`

Este método conta o valor de um *array* de cartas (*cartas_dealer* ou *cartas_player*) de acordo com as regras do blackjack. As cartas do 2 ao 9 pontuam com o valor marcado na carta (2 a 9 pontos). O Rei, a Dama e o Valete valem 10 pontos e o Às pode ter o valor de 1 ou 11 (o jogador escolhe como lhe dá mais jeito). Retorna os pontos resultantes.

d. `terminou()`

Este método recebe um *array* de cartas e verifica se a pontuação das cartas permitem terminar o jogo (rebentar ou 21) e se alguém ganhou. Retorna duas variáveis booleanas (Acabou ou não, ganhou ou não). Quem fizer 21 ganha imediatamente. Quem rebentar perde imediatamente.

e. `jogada_player()`

Este método vai buscar a próxima carta ao baralho e coloca-a no *array* de cartas do *player*. Utiliza o método `terminou()` para saber se as cartas do *player* permitem terminar ou ganhar o jogo. Retorna as duas variáveis booleanas do método `terminou()`.

f. `jogada_dealer()`

Este método vai buscar a próxima carta ao baralho e coloca-a no *array* de cartas do *dealer*. Utiliza o método `terminou()` para saber se as cartas do *dealer* permitem terminar (rebentar) ou ganhar (cartas tem o valor de 21) o jogo. Caso contrário, verifica se o *dealer* ganhou por ter mais pontos que o *player*. Retorna as duas variáveis booleanas do método `terminou()`. Se o *dealer* ganhou por ter mais pontos, antes de retornar as variáveis, ambas são colocadas a *true*.

3. Implemente as funções do ficheiro "blackjack_manager.js"
    a. `novo_jogo()`

       Esta função cria uma instância do objeto "blackJack", executa duas jogadas do *dealer* e uma do *player*. De seguida são atualizados o *dealer*, o *player* e são inicializados os botões da interface utilizador.

    b. `atualiza_dealer()`

       Esta função pede ao objeto "blackJack" as cartas do *dealer* e verifica se o *dealer* só tem 2 cartas para voltar a segunda (colocar um X). Depois começa a construir uma *string* para mostrar as cartas. A seguir, verifica o parâmetro de entrada (variável booleana "resultado"), de modo a acrescentar à *string* informação se o *dealer* ganhou ou perdeu. Depois atualiza a *string* no elemento HTML associado ao *dealer* e executa a função `finaliza_botoes()`.

    c. `atualiza_player()`

       Esta função pede ao objeto "blackJack" as cartas do *player* e começa a construir uma *string* para mostrar as cartas. A seguir, verifica o parâmetro de entrada, de modo a acrescentar à *string* informação se o utilizador ganhou ou perdeu. Depois atualiza a *string* no elemento HTML associado ao *player* e executa a função `finaliza_botoes()`.

    d. `jogada_dealer()`

       Esta função executa o método do objeto "blackJack" que realiza a jogada do *dealer* e atualiza o *dealer*.

    e. `jogada_player()`

       Esta função executa o método do objeto "blackJack" que realiza a jogada do *player* e atualiza o *player*.

    f. `dealer_acaba()`

       Esta função chama o método `terminou()` do objeto "blackJack" e atualiza o *dealer* com o resultado. Depois é criado um ciclo até que o *dealer* termine. Nesse ciclo, é realizada uma jogada do *dealer* e é atualizado o *dealer* em cada iteração.

4. Melhore a interface utilizador (opcional).