Como utilizar:

O jogo requer um cliente de SICSTUS Prolog a correr um servidor antes de se começar o jogo propriamente dito. Para correr este servidor, deve-se abrir um cliente SICSTUS, fazer consult ao ficheiro **`PrologServer/constructo3.pl`** e usar o comando "**`server.`**" (inclui o ponto). Após isto, não é necessário interagir mais com o servidor excepto para fechar a janela.

```
%   loading c:/program files (x86)/sicstus prolog vc12 4.3.2/library/codesio.po...
%   module codesio imported into user
%    module types imported into codesio
%    loading foreign resource c:/program files (x86)/sicstus prolog vc12 4.3.2/library/x86-win32-nt-4/codesio.dll in module codesio
%   loaded c:/program files (x86)/sicstus prolog vc12 4.3.2/library/codesio.po in module codesio, 0 msec 9296 bytes
* clauses for user:parse_input/2 are not together
* Approximate lines: 114-114, file: 'd:/programs/wamp/www/laigtp3/tp3/prologserver/server.pl'
%  consulted d:/programs/wamp/www/laigtp3/tp3/prologserver/server.pl in module user, 16 msec 111664 bytes
% consulted d:/programs/wamp/www/laigtp3/tp3/prologserver/constructo3.pl in module user, 47 msec 371128 bytes
yes
| ?- server.
Opened Server
```

Todo o jogo no browser controla-se com o rato por meio de cliques em elementos.

O Contructo – pode ser jogado em Android gratuitamente com o seguinte link – tem um objectivo simples. Os jogadores põem as suas peças no tabuleiro um de cada vez, e ganha que tiver mais torres com a sua cor no ápice. Só conta para a pontuação torres com pelo menos duas peças.

Há três tipos de peças circulares e cada jogador começa com três de cada. As maiores são um anel grande o suficiente para uma pequena lhe caber no interior do buraco. Um jogador tem peças brancas, o outro peças pretas. O tabuleiro em si consiste num grupo de 9 hexágonos sobre cada uma das quais as torres são contruidas. As casas pode-se dispor de qualquer forma desde que hajam adjacências indirectas entre todas.

A partir de fora do tabuleiro, as peças apenas podem se mover para casas vazias. A partir de uma casa só com uma peça, essa peça só pode se mover de forma a subir para o cume de uma torre que esteja numa casa adjacente à que se encontra. Peças em casas com mais que duas peças nunca se poderão voltar a mover.

As peças apenas podem subir numa torre por cima de uma peça que lhe seja mais pequena. Pequenas sobre as médias, Médias sobre as grandes. No entanto, ao mover-se no tabuleiro, uma peça grande pode ser colocada à volta de uma pequena que lhe era adjacente, formando uma torre de um só andar pertencente ao jogador da peça grande. Não é necessário haver uma peça pequena dentro da grande para uma peça média cobrir a grande, não à diferença nenhuma entre as torres do ponto de vista da pontuação.

Quando for a vez de um jogador que nao se pode mover mais, o jogo termina. Mesmo que o outro jogador se possa mover. Por norma, o jogador com maior numero de peças em casas só com essa peça é o perdedor.

Funções Importantes no código :

Ficheiro **reader/GameScene.js**. É o ficheiro da cena de jogo própriamente dito. Apesar de atingir múltiplas centenas de linhas de código, a grande maioria deve-se a funções herdadas dos trabalhos anteriores e à implementação dos ficheiros LSX. Funções de destaque:

- ***prototype.display()*** – Onde se encontram as chamadas de funções de display. O melhor local por onde começar a ver a parte gráfica do programa.
- ***prototype.GameDisplay()*** – Contem as funções de tranformação e display, de picking dos elementos que constroem a mesa, o tabuleiro e as peças do jogo.
- ***prototype. displayHUD()*** – Contem as funções de tranformação e display, de picking dos elementos 2D que aparecem no ecrã durante o jogo.
- ***prototype. displayMenu()*** – Contem as funções de tranformação e display, de picking dos menus que aparecem no principio do jogo.

Ficheiro **reader/GameState.js**. onde se encontra toda a informação do jogo e onde é chamada a maioria dos passos lógicos. Se qualquer alteração ocorrer no jogo, há grande hipótese senão mesmo certeza que esta ocorre aqui.

- ***Prototype.logic()*** – Uma máquina de estados para todas os momentos de jogo. É chamada durante o display da cena e chama todas as restantes funções do programa excepto os construtores.
- ***prototype.PickingLogic()*** – Dá inicio a outras funções, é responsável por escolher qual a acção a efectuar quando se fez um "picking" de um objecto com o rato.
- ***prototype.PieceMovementLogic()*** – Outro grade bloco importante, este muda as informações que o jogo usa para saber qual o estado de todas as peças e casas . Jogadas inválidas são permitidas por esta função, mas existem imensos avisos quando as regras são ignoradas que aparecem na consola.

Ficheiros a considerar:

Ficheiro **reader/primitives/GamePiece.js** e os seus derivados, **Small**, **Medium** e **Large** . Representam as peças do jogo. Reconhecem a sua cor, a sua localização no ambiente de jogo, e no tabuleiro própriamente dito.

Ficheiro **reader/primitives/HexagonPrism.js** e **Tabuleiro.js**. O Tabuleiro é o conjunto de todas as casas, cada uma delas representada por um hexagono de madeira. Um HexagonPrism, da mesma forma que as peças GamePiece, tem imensas referencias cruzadas com as anteriores, mas mais importante ainda, consegue dizer qual o estado da torre numa forma que o Prolog reconhece.

Ficheiro **reader/primitives/Table.js**. É a mesa que fica por baixo do tabuleiro.

Ficheiros **reader/shaders/select.frag** e **flat.vert** são os shaders utilizados pela cena para animar com cor as peças seleciondas. Ficheiros **reader/shaders/font.frag** e **font.vert** são os shaders utilizados para escrever a pontuação.

Ficheiros na pasta **PrologServer**. À exceção do ficheiro **server.pl**, todos estes são o jogo original tal e qual foi submetido no 1º projecto de PLOG.