Laboratório de Aplicações com Interface Gráfica

MIEIC – 2016/2017

Avaliação individual após Trabalho T1
(duração: 3:00 horas + 0:30 de tolerância)

O presente enunciado diz respeito a uma prova de avaliação prática da unidade curricular LAIG e contém um conjunto de três desenvolvimentos a efetuar sobre a base de trabalho desenvolvida no trabalho T1 das aulas práticas. Os computadores das salas reservadas para a prova encontram-se privados de rede. O único acesso permitido é ao software "SIGEX" para efeitos de *upload* dos trabalhos realizados (http://sigex.fe.up.pt; a validação é a normal do FEUPSIG).

Introduzir o "Código Público do Exame" correspondente:

- B207 (WINDOWS), Código Público do Exame: **ODX806**
- B208 (WINDOWS), Código Público do Exame: **REO352**
- B213 (WINDOWS), Código Público do Exame: **WEB770**
- B213 (LINUX), Código Público do Exame: **UCO565**

O resultado é uma página onde pode ser encontrada alguma informação assim como facilidades de *upload* de ficheiros.

Instruções para *Upload*

Siga cuidadosamente as seguintes instruções:

- Use sempre como base o **código original** do trabalho **T1** (não resolver um problema proposto sobre a solução do problema anterior).
- **Assinale no próprio código** as secções que alterou para resolver o problema, **com um ou mais comentários** do tipo (dependendo do problema que está a resolver):
  - //LAIGPROB1_inicio, //LAIGPROB1_fim;
  - //LAIGPROB2_inicio, //LAIGPROB2_fim;
  - //LAIGPROB3_inicio, //LAIGPROB3_fim.
- Para cada problema, crie um ficheiro de texto **ident.txt**, contendo:
  - Identificação do estudante (número e nome),
  - Identificação do Grupo,
  - Situação da resolução do problema em questão: "prob. completo", "não considera o detalhe xpto", "não resolvido"...
  - Algumas instruções julgadas pertinentes para o funcionamento do software, nomeadamente a forma de identificar o ficheiro "teste.dsx"
- Arquive os seguintes ficheiros num ficheiro .zip com o nome "prob1.zip"... "prob3.zip" (esta regra de nomeação deve ser ESTRITAMENTE seguida, **sem espaços nem extensões diferentes de .zip**):
  - o ficheiro **ident.txt**;
  - o **código-fonte**, incluindo o próprio **index.html** de arranque;
  - um ou mais **ficheiros DSX** necessários para demonstrar a resolução do problema em questão;
  - todas as **texturas** utilizadas.
- Cada problema, tendo em atenção o ponto anterior, deverá ser arquivado num diretório independente ("prob1",... "prob3") e cada um destes, comprimido em um ficheiro "prob1.zip", ... "prob3.zip".
- Cada ficheiro obtido "prob?.zip" deverá ser sujeito a *upload* utilizando as facilidades do SIGEX.

Notas:

- Para efeito de teste, garanta que, após arrancar o servidor, o *browser* usa o endereço 127.0.0.1:8080 (se necessário altere).
- Cada ficheiro "prob*.zip" deverá, isoladamente, ser sujeito a upload utilizando as facilidades do SIGEX.
- Caso não tenha resolvido um problema, deve submeter na mesma o respetivo ficheiro .zip, nem que contenha apenas o ficheiro **ident.txt** com a indicação de não-resolução. Isto é, no final, deve ter sempre três ficheiros submetidos: **prob1.zip, prob2.zip**, **prob3.zip**.
- Os identificadores finais dos arquivos serão construídos pelo sistema SIGEX, acrescentando-lhes um prefixo igual ao identificador do estudante respetivo.
- O sistema SIGEX permite eliminar ficheiros submetidos erradamente, e/ou submeter ficheiros com o mesmo nome, que serão sobrepostos aos existentes.

Enunciado - Novas funcionalidades sobre o trabalho T1

1. Nova primitiva Árvore (6 valores)

Crie um novo tipo de geometria "**`tree`**" que crie uma árvore constituída pelos seguintes componentes:

a)- Tronco: cone com altura ***th*** e raio de base ***tb***.

b)- Copa: ***nt*** triângulos isósceles rodados entre si de ângulos iguais (o primeiro dos quais coincidente com o plano ***xy***), segundo um eixo vertical coincidente com o eixo do tronco; cada triângulo possui dimensão de base ***cb*** e altura ***ch*** (***ch*** < ***th***, mensagem de erro em caso contrário); o topo da copa coincide com o topo do tronco.

Nota1: não haverá texturas para qualquer dos componentes da árvore; o material da copa será assumido pelo esquema normal de herança; o material do tronco ***tm*** será fornecido na instrução DSX

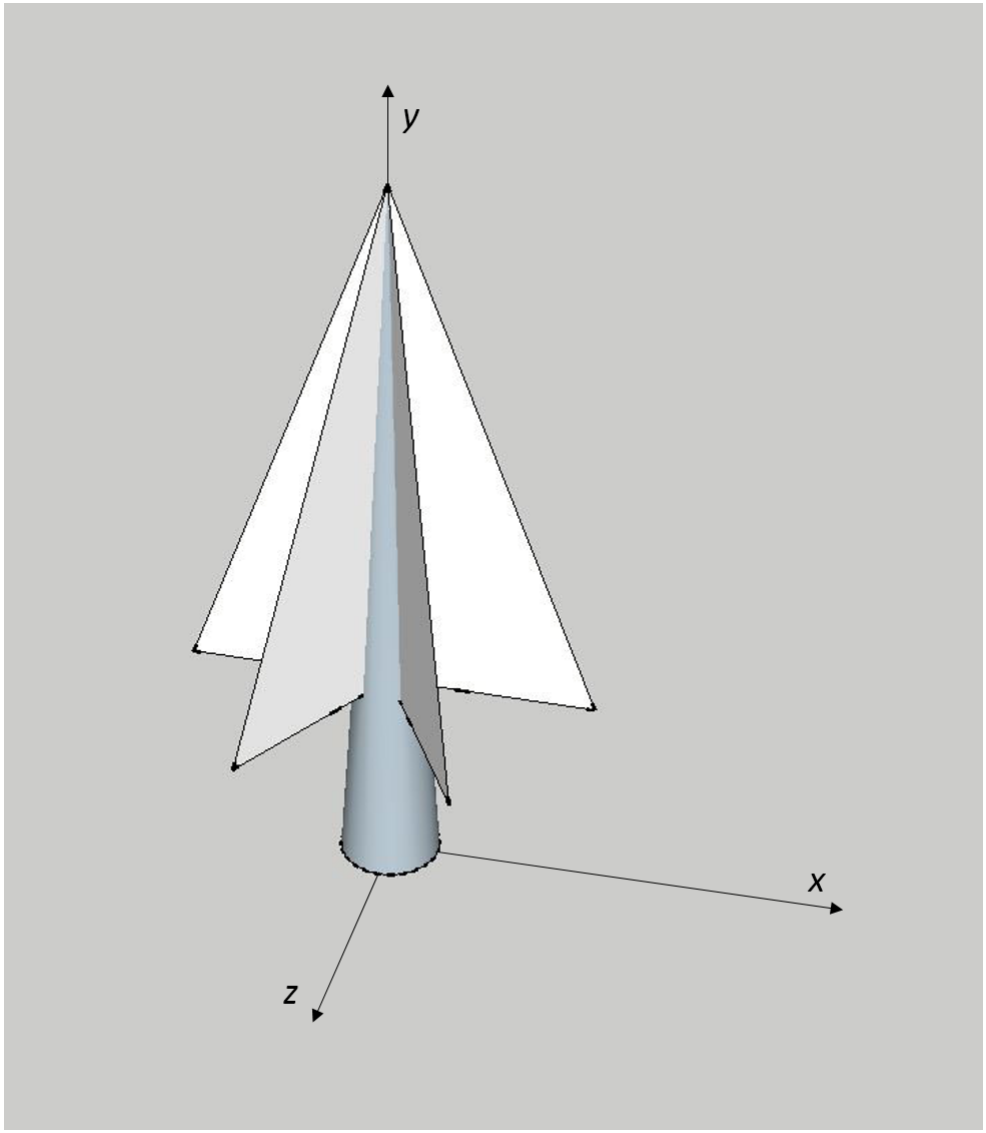
Nota2: os triângulos da copa devem ser visíveis de qualquer ponto de vista, independentemente da técnica back-face culling.

A sintaxe desta instrução, a colocar no bloco "**`primitives`**" é a seguinte:

```
<tree th="ff" tb="ff" ch="ff" cb="ff" nt="ii" tm="ss" />
```

Exemplo: a árvore da figura seria obtida pela instrução seguinte:

```
<tree th="14.0" tb="1.0" ch="10.0" cb="8.0" nt="3" tm="tmater" />
```



2. Inversão de componentes dos materiais (7 valores)

Pretende-se que, na referência a uma material no contexto de um nó, possa ser especificado um atributo *invert*, **opcional**, do tipo *Booleano*, que, quando declarado com o valor "***true***", substitui todos os valores de ***RGB*** das suas componentes pelos seus complementos para um, ou seja, um valor **0.8** é substituído por **0.2**; **0.3** é substituído por **0.7**. Nota: as componentes *a*/*alpha* e *shininess* devem manter-se inalteradas.

Esta inversão aplica-se a qualquer material, seja declarado no nó ou recebido por *inherit*. No caso de existência de lista de materiais permutáveis com a tecla ***m***/***M***, alguns podem ter *invert* e outros não.

```
<material id="ss" invert="tt" />
```

3. Escalamento interativo de nós (7 valores)

Na declaração de transformações de um nó passa a existir a declaração opcional de um novo tipo de transformação - INTERACT - descrito abaixo. Caso seja usado este tipo de transformação num nó, **não deve ser declarada mais nenhuma transformação ou referência a transformações**.

```
<interact scale="ff" />
```

O elemento INTERACT incorpora a capacidade de interagir com um nó, sendo que neste exercício pretende-se apenas a capacidade de aplicar um escalamento com base no atributo *scale* em função da interação do

utilizador. Ao premir a tecla ***x/X***,o conteúdo do nó respetivo (e dos seus descendentes) é escalado nas três dimensões segundo o fator *scale*; ao premir a tecla ***z/Z***, o conteúdo é escalado segundo o fator inverso (*1/scale*); ao premir a tecla ***c/C***, o conteúdo assume um escalamento de **1.0** (o equivalente a não ser aplicado nenhum fator de escala, que deve ser também o comportamento inicial no arranque da aplicação). O efeito da teclas ***x/X*** e ***y/Y*** é sempre verificado em relação ao tamanho original do objeto, independentemente de se ter ou não pressionado anteriormente a outra tecla**.**

Por exemplo, se:

```
<interact scale="1.2" />
```

Ao premir a tecla ***x/X***,o conteúdo é escalado nas três dimensões em **120%**; ao premir, depois, a tecla ***z/Z***, o conteúdo é escalado segundo **83.33%**. Voltando a pressionar ***x/X***, o escalamento do objecto volta a ser **120%**. Precionando c/C, o escalamento volta a tomar o seu valor original, ou seja, **100%**.

Podem existir diferentes nós com este tipo de transformação, e com fatores diferentes, sendo todos aplicados aos respetivos nós quando as teclas ***x/X***, ***y/Y*** e ***c/C*** são pressionadas.

O fator de escala não é cumulativo, ou seja, se ***x/X*** for pressionado duas ou mais vezes consecutivas, o fator de escala correspondente só é aplicado uma vez. No entanto, um nó pode pode ser afetado por um escalamento recebido por descendência e por outro declarado em si próprio.

Porto e FEUP, 9 de novembro de 2016

A. Augusto de Sousa,
Jorge Barbosa,
Rui Rodrigues,
Alexandre Carvalho

Resumo da Linguagem DSX modificada

Nota: as alterações à especificação inicial encontram-se marcadas a ***vermelho, bold, itálico.***

Um documento DSX divide-se em blocos, cada um iniciando-se com um termo identificador de bloco, implementado em XML na forma de uma *tag*. A referência a uma *tag* inicia-se com um identificador alfanumérico entre os dois caracteres "< >" (por exemplo `<lights>`) e terminando com o mesmo identificador antecedido de uma barra de divisão (no mesmo exemplo, `</lights>`); entre as duas ocorrências, descreve-se o conteúdo do elemento identificado pela *tag*. A sequência de blocos é a seguinte:

```
scene               <!-- global values -->
views               <!-- specification of all views -->
illumination        <!-- illumination parameters -->
lights              <!-- specification of all light sources -->
textures            <!-- specification of all textures -->
materials           <!-- specification of all materials -->
transformations     <!-- specification of geometric transformations →
                    <!-- for use in different components -->
primitives          <!-- specification of all primitives -->
components          <!-- specification of all components: objects  -->
                    <!-- composed of primitives and other components -->
```

Os últimos seis blocos anteriores contêm definições de várias entidades do tipo correspondente (várias luzes, várias texturas, etc...). Cada uma dessas entidades contém um identificador do tipo *string*. Cada identificador deve ser único dentro de cada bloco (por exemplo, não podem existir duas fontes de luz com o mesmo identificador).

Uma listagem representativa da sintaxe pretendida pode ser encontrada abaixo:

```
<!-- Os comentarios devem ter espacos no inicio e no fim, a -->
<!-- separar dos hifens -->
<!-- Nao devem ser usados caracteres especiais (p.ex. acentos) -->
<!-- Todas as tags e atributos sao obrigatorios, exceto onde for -->
<!-- referido o contrario -->

<!-- Na descricao abaixo, os simbolos utilizados tem o seguinte significado: -->
    <!-- ii: integer value -->
    <!-- ff: float value -->
    <!-- ss: string value -->
    <!-- cc: character "x" or "y" or "z" -->
    <!-- tt: "0" or "1" with Boolean significance -->


<dsx>


    <!-- deve definir-se um objeto para raiz do grafo , assim -->
    <!-- como o comprimento dos tres eixos (cilindros) -->
    <scene root="ss" axis_length="ff" />


    <views default="ss" >

        <!-- declaracao obrigatoria de pelo menos uma vista; -->
        <!-- se varias vistas declaradas, o default e' a -->
        <!-- primeira vista; de cada vez que se pressione a tecla v/V, -->
        <!-- a vista muda para a proxima da lista; da -->
        <!-- ultima vista da lista volta 'a primeira -->
        <perspective id="ss" near="ff" far="ff" angle="ff">
            <from x="ff" y="ff" z="ff" />
            <to x="ff" y="ff" z="ff" />
        </perspective>

    </views>


    <illumination doublesided="tt" local="tt" >

        <ambient r="ff" g="ff" b="ff" a="ff" />
        <background r="ff" g="ff" b="ff" a="ff" />

    </illumination>


    <lights>

        <!-- Deve existir um ou mais blocos "omni" ou "spot" -->
        <!-- Os identificadores "id" nao podem ser repetidos -->
        <omni id="ss" enabled="tt" >
            <location x="ff" y="ff" z="ff" w="ff" />
            <ambient r="ff" g="ff" b="ff" a="ff" />
            <diffuse r="ff" g="ff" b="ff" a="ff" />
            <specular r="ff" g="ff" b="ff" a="ff" />
        </omni>

        <spot id="ss" enabled="tt" angle="ff" exponent="ff">
            <!-- atencao, "target" e' diferente de "direction" -->
            <target x="ff" y="ff" z="ff" />
            <location x="ff" y="ff" z="ff" />
            <ambient r="ff" g="ff" b="ff" a="ff" />
            <diffuse r="ff" g="ff" b="ff" a="ff" />
            <specular r="ff" g="ff" b="ff" a="ff" />
        </spot>
    </lights>


    <textures>
```

```
        <!-- Deve existir um ou mais blocos "texture" -->
        <!-- Os identificadores "id" nao podem ser repetidos -->
        <!-- length_s e length_t sao fatores de escala de textura:-->
        <!-- Exemplo length_s=3.0: uma ocorrencia da textura, em -->
        <!-- comprimento, deve cobrir um comprimento igual -->
        <!-- a 3 unidades;  -->
        <!-- Exemplo length_t=0.4, uma ocorrencia da textura, em -->
        <!-- largura, deve cobrir uma largura igual a 0.4 unidades.  -->
        <!-- Transf. Geometr. do tipo escalamento sobre os -->
        <!-- objetos respetivos podem a violar esta regra.         -->
        <!-- Nao necessario aplicar fatores de escala em  -->
        <!-- quadricas (esfera, ciclindro...) -->
        <texture id="ss" file="ss" length_s="ff" length_t="ff" />

    </textures>


    <materials>

        <!-- Deve existir um ou mais blocos "material" -->
        <!-- Os identificadores "id" nao podem ser repetidos -->
        <material id="ss" >
            <emission r="ff" g="ff" b="ff" a="ff" />
            <ambient r="ff" g="ff" b="ff" a="ff" />
            <diffuse r="ff" g="ff" b="ff" a="ff" />
            <specular r="ff" g="ff" b="ff" a="ff" />
            <shininess value="ff" />
        </material>

    </materials>


    <transformations>

        <!-- Deve existir um ou mais blocos "transformation" -->
        <!-- Os identificadores "id" nao podem ser repetidos -->
        <transformation id="ss">
           <!-- deve existir pelo menos uma instrução de transformacao; -->
           <!-- podem ser usadas tantas instrucoes quantas as -->
           <!-- necessárias, mas escritas pela mesma ordem que
           <!-- seria usada na sua escrita diretamente em código WebGL -->

           <translate x="ff" y="ff" z="ff" />
           <rotate axis="cc" angle="ff" />
           <scale x="ff" y="ff" z="ff" />
        </transformation>

    </transformations>


    <primitives>

        <!-- Deve existir um ou mais blocos "primitive" -->
        <!-- Os identificadores "id" nao podem ser repetidos -->
        <primitive id="ss">

            <!-- apenas pode existir UMA das seguintes tags: -->
            <!--     rectangle, triangle, cylinder, sphere, torus -->
            <!-- os parametros devem ser interpretados, genericamente como-->
            <!-- em OpenGL/GLUT; o cilindro deve adicionalmente ter tampas -->
            <rectangle x1="ff" y1="ff" x2="ff" y2="ff" />
            <triangle x1="ff" y1="ff" z1="ff" x2="ff" y2="ff" z2="ff" x3="ff" y3="ff"
z3="ff" />
            <cylinder base="ff" top="ff" height="ff" slices="ii" stacks="ii" />
            <sphere radius="ff" slices="ii" stacks="ii" />
            <torus inner="ff" outer="ff" slices="ii" loops="ii" />

            <tree th="ff" tb="ff" ch="ff" cb="ff" nt="ii" tm="ss" />
```

```
        </primitive>

    </primitives>


    <components>

        <component id="ss">

            <!-- bloco "transformation" obrigatorio -->
            <transformation>

                <!-- deve conter uma referencia a uma das "transformation" -->
                <!-- declaradas anteriormente -->
                <transformationref id="ss" />

                <!-- ou, ALTERNATIVAMENTE, transformacoes explicitas, -->
                <!-- usando zero ou mais instrucoes como as seguintes,  -->
                <!-- seguindo as mesmas regras que no bloco transformations-->
                <translate x="ff" y="ff" z="ff" />
                <rotate axis="cc" angle="ff" />
                <scale x="ff" y="ff" z="ff" />

                <!-- ou, ALTERNATIVAMENTE, uma transformacao interativa, -->
                <interact  scale="ff" />

            </transformation>


            <!-- declaracao obrigatoria de pelo menos um material; -->
            <!-- o material id="inherit", mantem (herda) material do "pai" -->
            <!-- se varios materiais declarados, o default e' o -->
            <!-- primeiro material; de cada vez que se pressione a tecla m/M, -->
            <!-- o material muda para o proximo material da lista; do -->
            <!-- ultimo material da lista volta ao primeiro -->
            <materials>
                <!-- novo atributo "invert", opcional -->
                <material id="ss" invert="tt" />
            </materials>

            <!-- declaracao obrigatoria de texture -->
            <!-- id="inherit" mantem (herda) a textura do objecto "pai" -->
            <!-- id="none" remove a textura recebida do pai -->
            <!-- a textura declarada sobrepoe a textura recebida do -->
            <!-- objecto "pai" -->
            <texture id="ss" />

            <!-- bloco "children" obrigatorio num "component" -->
            <children>

                <!-- deve existir uma ou mais tags "componentref" e/ou -->
                <!-- "primitiveref", identificando outros -->
                <!-- componentes ou primitivas -->
                <componentref id="ss" />
                <primitiveref id="ss" />

            </children>

        </component>

    </components>

</dsx>
```