Computação Gráfica

Primeiro trabalho, Semestre de Inverno de 2013/2014

Neste trabalho os alunos desenvolverão competências no desenvolvimento de aplicações 2d interactivas que lidam com a rasterização de linhas e círculos, assim como desenho e preenchimento de polígonos. Tomam ainda contacto com transformações geométricas 2d.

Desenvolva uma aplicação 2D interactiva, ***CGPaint***, em Java, para desenho, preenchimento e manipulação de formas geométricas. Com esta aplicação deve ser possível desenhar formas geométricas (linhas, triângulos, polígonos) numa área de desenho usando cores de contorno e preenchimento definidas pelo utilizador através de controlo gráfico criado para o efeito. Após desenho, as formas podem ser seleccionadas e editadas no que diz respeito às suas cores e transformação geométrica (translação, rotação e escala) na área de desenho.

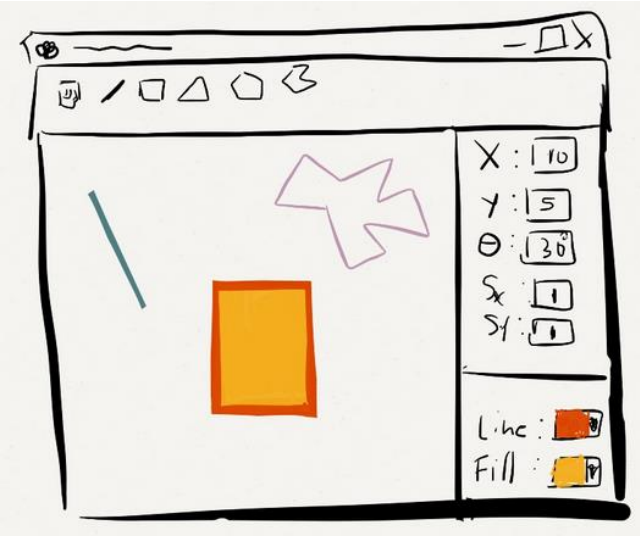


Figura 1: Esboço da aplicação a desenvolver

De seguida são apresentadas as funcionalidades obrigatórias a implementar na aplicação.

Parte 1: Desenho livre

Na parte 1 será desenvolvida a primeira versão da aplicação, onde é possível fazer desenho livre, assim como limpar a área de desenho.

1. Estude a **área de desenho** da aplicação, analisando a aplicação de exemplo fornecida em anexo ao trabalho que exemplifica a implementação de uma área de desenho com as características indicadas. A listagem de código seguinte mostra as interfaces `IRasterDevice` que representa um dispositivo capaz de gerar imagens

raster e a interface `ITouchDevice` que representa as caracteristicas de interacção Homem-máquina (i.e. mouse) do dispositivo.

```
public interface IRasterDevice {
     Component getComponent();
     void addRenderListener(IRenderListener renderListener);
}
public interface IRenderListener {
     void onRender(ITouchRasterDevice rasterDevice);
}
public interface IRenderContext {
     void setPixel(int x, int y, Color color);
}

public interface ITouchDevice {
     void addTouchListener(ITouchListener touchListener);
}
public interface ITouchListener {
     void onMove(int x, int y);
     void onClick(int x, int y, int button);
     void onDrag(int x, int y, int button);
}
```

2. Implemente o **modo de desenho livre**, em que são desenhados *pixeis* na área de desenho. Este modo deve ser activado através de um botão (com ícone ilustrativo da operação[1]) numa barra de tarefas da aplicação. A operação também deve estar disponível num menu (e.g. `Drawing Mode`) da aplicação.

   No modo livre devem ser desenhados *pixeis* durante o 'clique e arrasto' do rato por cima da área de desenho. Nesta fase deve desenhar os *pixeis* a vermelho.

3. Implemente a acção de **limpeza da área de desenho** em que a área de desenho deve ser limpa de forma a ficar com a cor branca.

Parte 2: Rasterização de primitivas 2d

Na parte 2 será adicionado suporte para desenho de primitivas gráficas 2d, tais como desenho de linhas, triângulos, círculos e polígonos.

1. Implemente o **modo de desenho de linhas**. Neste modo devem ser desenhadas linhas a cada dois cliques do rato na área de desenho. O primeiro clique representa o ponto de origem e o segundo ponto representa o ponto de destino. Deve usar o algoritmo de rasterização de linhas de *bresenham* apresentado na aula.

2. Crie, ou use, um controlo gráfico *swing*, para **escolha de cor**. A aplicação deve incluir dois destes controlos, para a definição da cor de contorno (linhas e pontos) e

[1] Sugere-se a utilização do pacote de *icons* glyphicons (http://glyphicons.com/).

cor de preenchimento. Actualize o código desenvolvido para tirar partido desta nova funcionalidade.

3. Implemente o **modo de desenho de círculos**. Neste modo o primeiro clique identifica o centro do círculo e o segundo clique um ponto do seu contorno (ou o seu raio). Deve usar o algoritmo de *Mid Point* para desenhar o contorno do círculo. Não é necessário preencher o círculo.

4. Implemente o **modo de desenho de polígonos convexos** que usam a cor de contorno no contorno do polígono e a cor de preenchimento no interior. O polígono é definido através de cliques consecutivos do rato sobre a área de desenho terminando ao clicar com o botão direito (de contexto) do rato. A implementação pode assumir que o utilizador definiu os vértices usando o sentido contrário ao dos ponteiros do relógio.

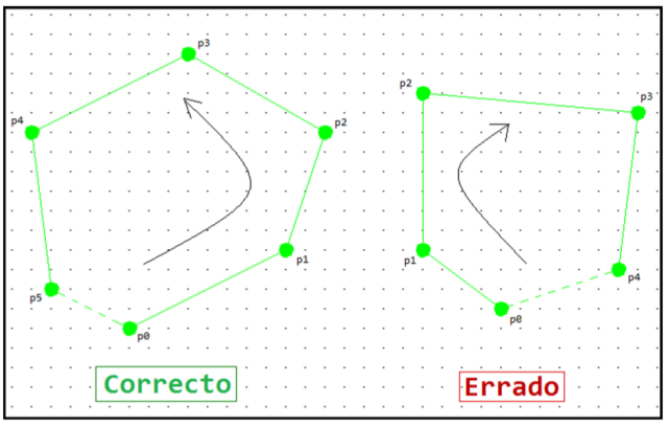


Figura 2: Desenho de polígonos no sentido contrário ao ponteiro do relógio (CCW)

5. Implemente o **modo de desenho de triângulos**. O triângulo é definido a cada três cliques do rato na área de desenho. Neste modo o interior do triângulo deve ser preenchido usando o algoritmo de sombreamento de *Gouraud*, e as cores dos vértices são vermelho, verde e azul. O contorno do triângulo não deve ser desenhado.

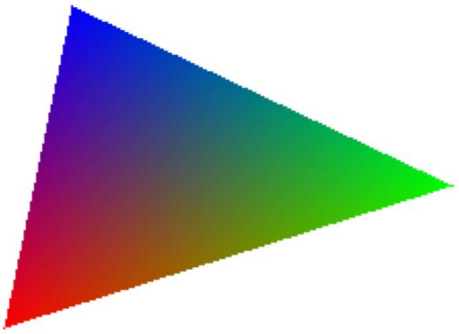


Figura 3: Triângulo preenchido com algoritmo de Gouraud

Parte 3: Modo de selecção e transformações geométricas

A última parte tem como objectivo adicionar suporte para manipulação das formas geométricas adicionadas anteriormente. Após selecção deve ser possível alterar a posição, rotação e escala das formas.

1. Implemente o **modo de selecção** que deve ser activado através de opção na barra de tarefas. Após selecção, usando clique do rato sobre a forma, deve ser dado destaque aos elementos representativos da forma, tais como pontos que a representam e centro da forma (importante para a operação de rotação).

2. Implemente as operações de **translação, escala e rotação** sobre a forma seleccionada. Pode usar manipulação directa da forma, e/ou controlos (e.g. caixas de texto) de edição. Note que nesta fase é essencial que representação interna de uma forma geométrica tenha associada uma (ou mais) matrizes de transformação.

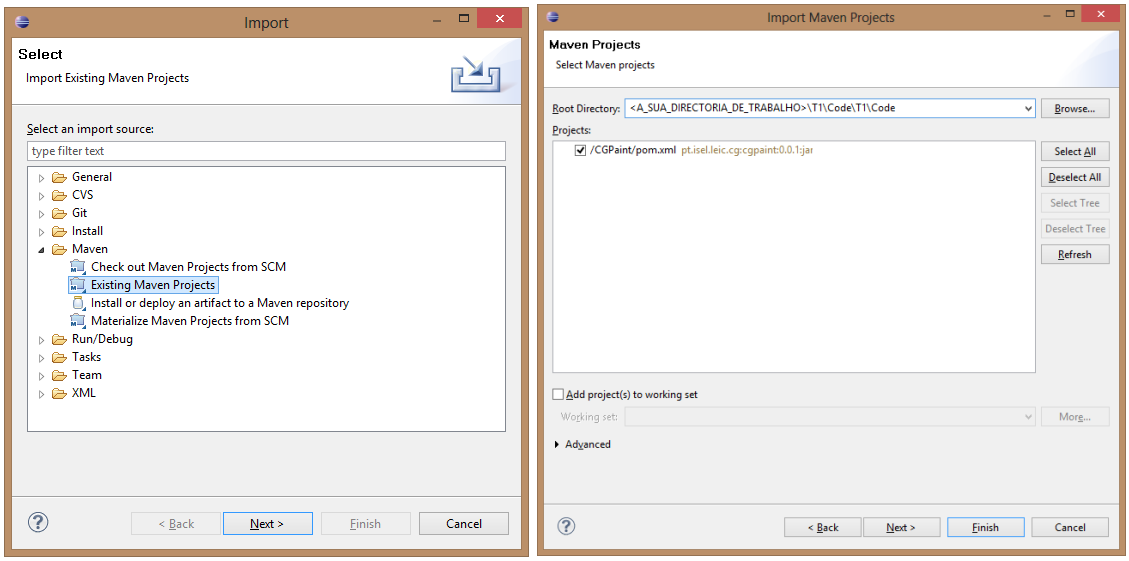
Código base para realização do trabalho

O código base para realização deste trabalho está disponível no repositório de código git da turma no GitHub (https://github.com/isel-leic-cg/1314i-li31d_li51d-public), na pasta `T1/Code`.

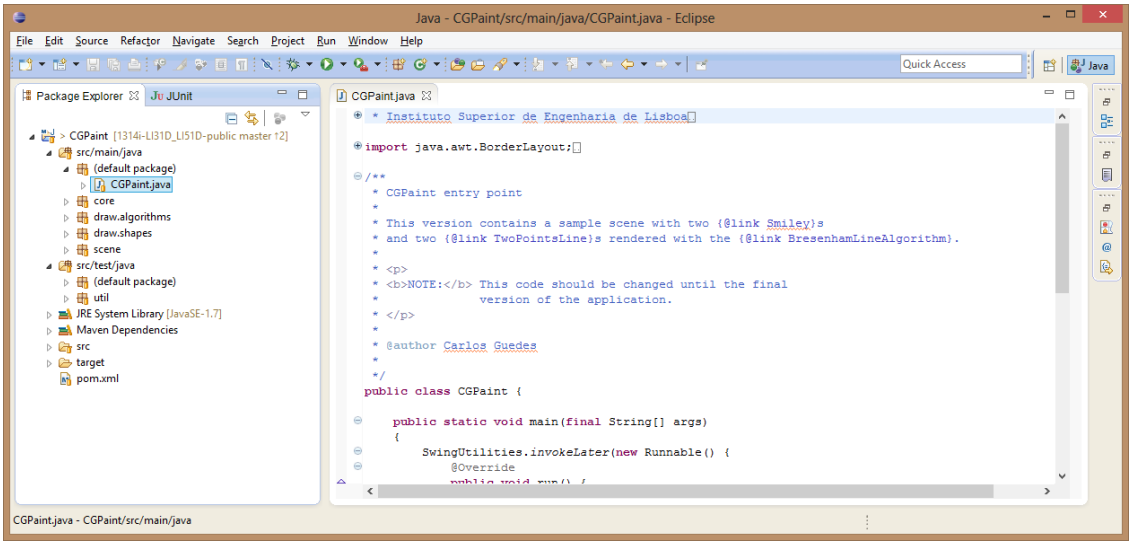
O projecto usa a estrutra de ficheiros maven (http://maven.apache.org/), e é composto por: aplicação a desenvolver (`src/main/java`); e código de teste (`src/test/java`).

Para compilar a aplicação pela linha de comandos, depois de instalar o maven, deve executar `mvn package`. Após sucess, execute a aplicação localizada em `target\cgpaint-0.0.1.jar`.

Se usar o ambiente de desenvolvimento eclipse (http://www.eclipse.org/downloads/), sugere-se crie um *workspace* em (`T1/Code`) e use a opção `File -> Import` e depois "Existing Maven Projects" tal como indicado nas figuras seguintes.



Pode comprovar que a importação correu bem se ficou com o projecto num estado parecido ao da seguinte figura.



A avaliação do trabalho tem em conta todos os aspectos do *software* desenvolvido, nomeadamente: cumprimento de requisitos, estruturação, qualidade e documentação do código produzido.

A entrega deste trabalho consiste na entrega de um **ZIP com o código desenvolvido** e por um **documento** (sem capa, índice, resumo, etc..) com considerações sobre o trabalho desenvolvido. Este documento deve ter o máximo de duas folhas. O documento deve incluir um cabeçalho, tal como neste enunciado, com identificação do grupo e respectivos membros.

Bom trabalho