visionlab
visualization laboratory

Pontifícia Universidade Católica
do Rio de Janeiro

Introdução a Programação de Jogos

Aula 08 – Utilizando Sprites e Áudio na PlayLib

Prof. Augusto Baffa
< abaffa@inf.puc-rio.br>

Biblioteca Gráfica - PlayLib

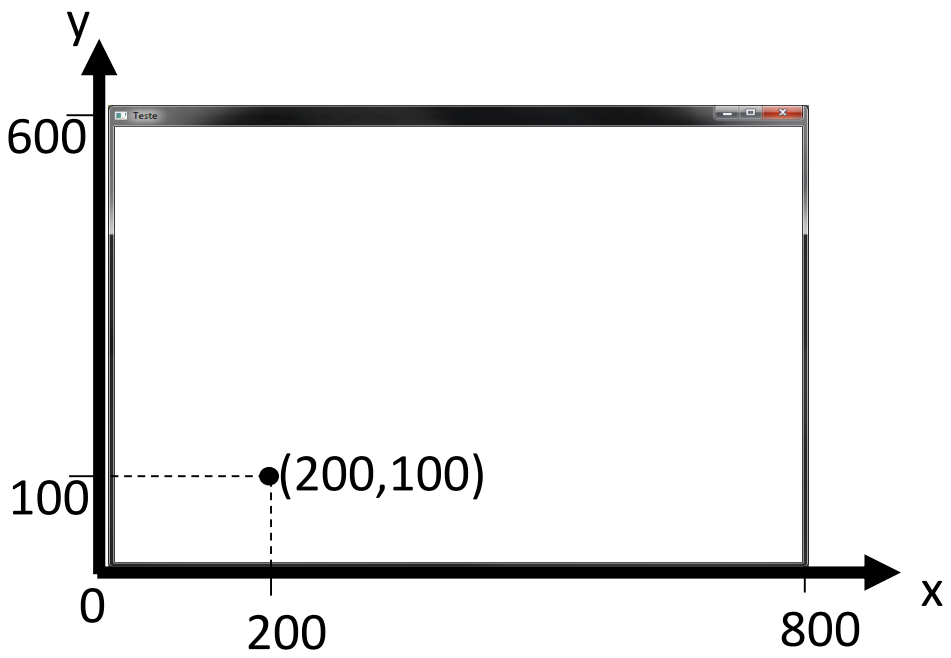
- **Conjunto de funções** para criação e manipulação de formas geométricas, imagens, áudio, janelas...
- Baseada na API **OpenGL**.
- Pode ser usada para criação de **jogos 2D**, **simulações**, **animações** e outros aplicativos.
- **Desenvolvida especialmente para esse curso!**

Estruturas

Struct são coleções de dados heterogêneos agrupados em uma mesma estrutura de Dados

Exemplo:

Armazenar as coordenadas (x,y) de um ponto

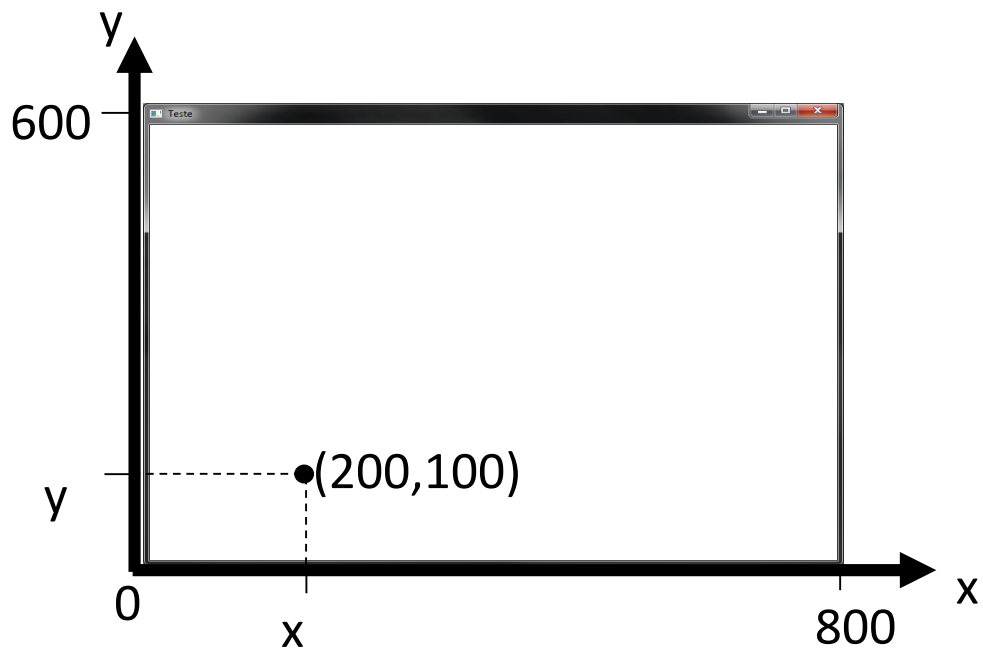


Estruturas

- Declaração:

```
struct coordenada{
        int x;
        int y;
};

coordenada p1, p2;
```



- A estrutura “coordenada” contém dois inteiros, x e y
- p1 e p2 são duas variáveis tipo coordenada.

Declaração de Estrutura

- Formato da declaração:

```
struct nome_da_estrutura {
        tipo_1 dado_1;
        tipo_2 dado_2;
        ...
        tipo_n dado_n;
};
```

- A estrutura pode agrupar um número arbitrário de dados de tipos diferentes

- Pode-se nomear a estrutura para referenciá-la

Estruturas

- Acesso aos dados:

variável_struct.campo

- Ex:

```
p1.x = 10;
p1.y = 20;
p2.x = 15;
p2.y = 15;

if (p1.x >= p2.x) &&
            (p1.y >= p2.y) ...
```

Exemplo de Estrutura

- Exemplo:

```
typedef struct train
{
        float x;                //posicao x do trem
        float y;                //posicao y do trem
        int animation_clip;  //clipa da animacao do trem
        int animation_index;            //indice atual da animacao do trem
        float time_next_frame; //contador de tempo para avancar na animacao
        bool stoped;            //indicador de trem parado
        float speed;            //velocidade do trem
        bool light_active;    //indicador de luz ativa do trem
        float angle;                    //angulo do trem
        int track_sector;      //setor atual do trem na pista
} Train;

Train train;
```

Exemplo de Estrutura

- Exemplo 2:

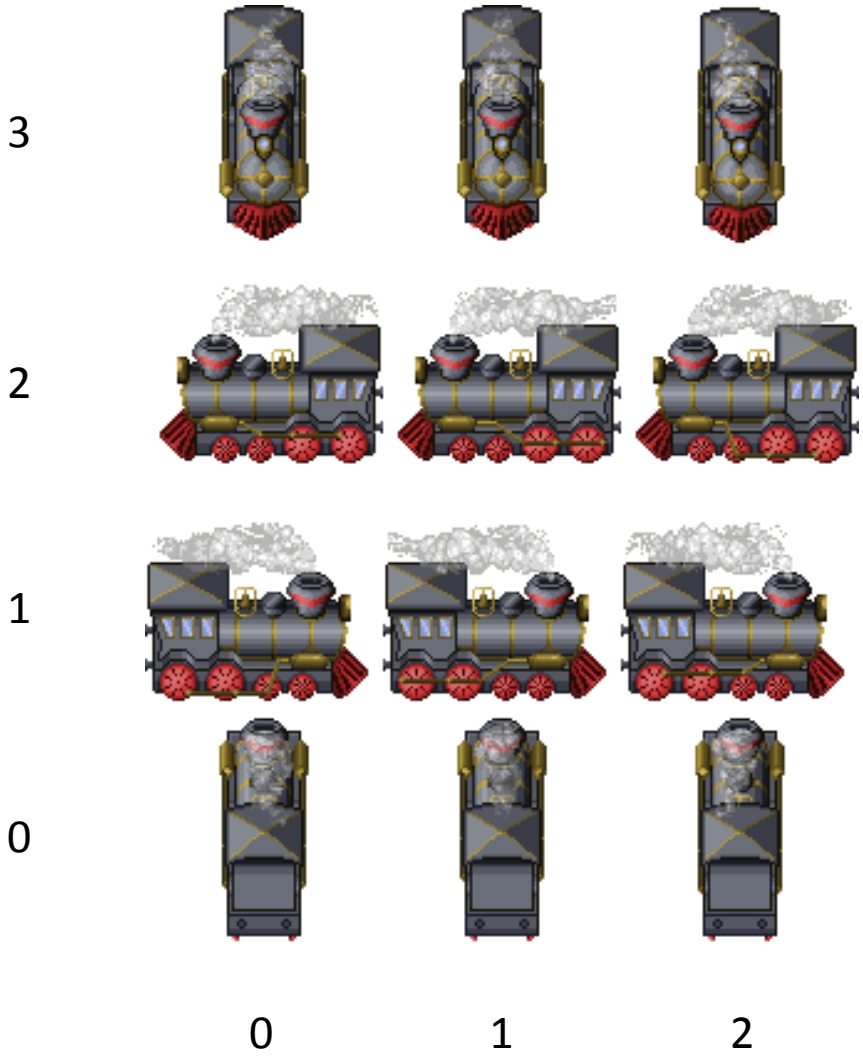
```
//Estrutura para armazenar os dados do sprite do trem

typedef struct sprite{
        int x;
        int y;
        int width;
        int height;
} Sprite;


//Variaveis para armazenar os sprites e imagens usadas na aplicacao

Sprite sprite[4][4];
```

Carregando Sprites com Struct



Carregando Sprites com Struct

- **Exemplo de carga de Sprites:**

```
#define SPRITE_SIZE 96

Sprite sprite[4][3];
Image sprite_sheet;

                                                    …

int main(void)
{
  ...
  sprite_sheet.LoadPNGImage("Train3.png");
  InitSprites();
  ...
}
```

Carregando Sprites com Struct

…

```
void InitSprites()
{
   int x = 0, y = 0;
   int spritex = 0;
   int spritey = 0;

   for (int y = 0; y < 3; y++)
      for (int x = 0; x < 4; x++)
      {
         sprite[y][x].x = 0 + SPRITE_SIZE * x;
         sprite[y][x].y = 0 + SPRITE_SIZE * y;
         sprite[y][x].width = SPRITE_SIZE;
         sprite[y][x].height = SPRITE_SIZE;
      }
}
```

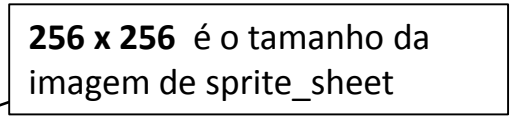
…

Utilizando as variáveis de Estrutura

```
int main(void)
{
   ...
//Inicializa as variaveis de controle do trem
train.x = 0;
train.y = 0;
train.animation_clip = 3;
train.animation_index = 0;
train.time_next_frame = 0;
train.stoped = true;
train.speed = 0;
train.light_active = false;
train.angle = -1;
train.track_sector = 1;
…
}
```

Utilizando as variáveis de Estrutura

- Desenhando o Trem:




```
graphics.DrawImage2D(train.x, train.y, 256, 256,
sprite[train.animation_clip][train.animation_index].x,
sprite[train.animation_clip][train.animation_index].y,
sprite[train.animation_clip][train.animation_index].width,
sprite[train.animation_clip][train.animation_index].height, sprite_sheet);
```

Executando um arquivo de áudio

- **Para executar um arquivo de audio é necessário:**
  - **(1)** Criar uma variável do tipo **Audio**.
  - **(2)** Carregar o arquivo usando o comando **LoadAudio**.
  - **(3)** Executar efetivamente o áudio no dispositivo de som usando o comando **Play**.

Executando um arquivo de áudio

- **(1) Criar uma variável do tipo Audio:**

```
Audio musica1;
```

OBS: Sempre declare as variáveis Audio como **variáveis globais**.

Exemplo:

```
#include "Graphics.h"
#include "Audio.h"

Graphics graphics;
Audio musica1;
Audio musica2;

int main(void)
{
...
```


Variáveis Audio declaradas no inicio do programa. Antes e fora da função principal ou outras funções.

Executando um arquivo de áudio

- **(2) Carregar o arquivo usando o comando LoadAudio:**

```
musica1.LoadAudio("Musica.mp3");
```

Exemplo:

```
int main(void)
{
...
  musica1.LoadAudio("Musica.mp3");
...
}
```

Carrega o arquivo **Musica.mp3** para a variável musica1.

OBS: Cada arquivo de áudio deve ser carregado **apenas uma vez**. Por isso, nunca carregue o áudio diretamente de dentro do Loop Principal.

Executando um arquivo de áudio

- **(3) Executar efetivamente o áudio no dispositivo de som usando o comando Play.**

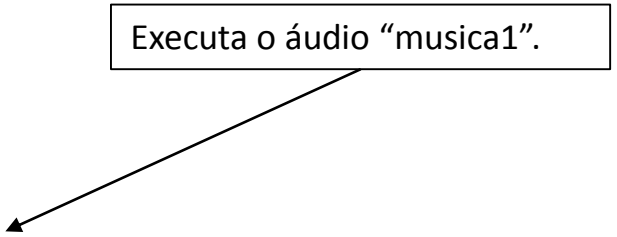
```
musica1.Play();
```

Exemplo:

```
Audio musica1;

int main(void)
{
  musica1.LoadAudio("Musica.mp3");
  ...
  musica1.Play();
  ...

}
```



Executando um arquivo de áudio

- **Observações importantes sobre arquivos de áudio:**
  - **Somente são aceitas imagens nos formatos WAV ou MP3**.
  - Cerifique-se de que as imagens que serão lidas estão **dentro da pasta do seu projeto do Visual Studio**. Se preferir armazena-las em outras pastas você deve fornecer o caminho completo para o diretório onde os arquivos de áudio estão para o comando LoadAudio.
  - Se o seu arquivo de áudio estiver em **outro formado** (3GP, OGG, WMA...) você deve convertê-la para o formato WAV/MP3 antes de carrega-lo.

Executando um arquivo de áudio

- **Outras funções para o tipo Audio:**
  - Para **encerrar a execução** um arquivo de áudio utilizamos a função **Stop()**:

```
musica1.Stop();
```

  - Podemos interromper temporariamente a execução de um arquivo de áudio utilizando o comando **Pause()** e retomá-lo utilizando o commando **Play()**. Neste caso o arquivo de áudio **continua a execução a partir do ponto em que foi interrompido**.

```
musica1.Pause();
…
musica1.Play();
```

Executando um arquivo de áudio

- **Outras funções para o tipo Audio:**
  - Para **verificarmos** se o arquivo **de áudio está em execução**, consultamos a função **IsPlaying()** que retorna verdadeiro em caso afirmativo.

musica1.IsPlaying()

Exemplo:

```
if(musica1.IsPlaying())
    . . .
else
    . . .
```

Exercícios

Lista de Exercícios 07 – Sprite Sheets e Audio

http://www.inf.puc-rio.br/~abaffa/iue1503/