Pontifícia Universidade Católica
do Rio de Janeiro

Introdução a Programação de Jogos

Aula 12 – Interação na PlayLib

Edirlei Soares de Lima
<elima@inf.puc-rio.br>

visionlab
visualization laboratory

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!**

visionlab
visualization laboratory

Tratando Entradas do Teclado

- Para poder tratar os eventos gerados pelo teclado (**teclas sendo pressionadas**) é necessário criar uma função para essa tarefa.

- Essa função deve ter a seguinte sintaxe:

```
void KeyboardInput(int key, int state, int x, int y))
{
   /* Bloco de Comandos */
}
```

- Também é **necessário indicar** que essa é a sua função para tratar eventos de teclado:

```
graphics.SetKeyboardInput(KeyboardInput);
```

visionlab
visualization laboratory

Tratando Entradas do Teclado

- Dessa forma, sempre que uma tecla normal do teclado for pressionada a função **KeyboardInput** será executada e o parâmetro key indicará qual tecla foi pressionada. O parâmetro state indicará se a tecla foi pressionada ou liberada.
- **Exemplo:**

```
void KeyboardInput(int key, int state, int x, int y))
{
  if ((key == 'f')&&(state == KEY_STATE_DOWN))
  {
    graphics.SetFullscreen(true);
  }
  if ((key == KEY_RIGHT)&&(state == KEY_STATE_DOWN))
  {
    posicao_personagem_x = posicao_personagem_x + 2;
  }
  if ((key == KEY_ESC)&&(state == KEY_STATE_DOWN))
  {
    exit(0);
  }
}
```

Se a letra f for pressionada

Coloca o programa em tela cheia

Se a seta da direita for pressionada

Incrementa em +2 uma variável que representa a posição de um personagem

Se a tecla esc for pressionada

Fecha o programa

visionlab
visualization laboratory

Códigos das Teclas Especiais

- KEY_LEFT
- KEY_UP
- KEY_RIGHT
- KEY_DOWN
- KEY_PAGE_UP
- KEY_PAGE_DOWN
- KEY_HOME
- KEY_END
- KEY_INSERT
- KEY_ESC
- KEY_ENTER
- KEY_BACKSPACE
- KEY_LEFTCTRL
- KEY_RIGHTCTRL
- KEY_LEFTSHIFT
- KEY_RIGHTSHIFT
- KEY_LEFTALT
- KEY_RIGHTALT
- KEY_TAB
- KEY_F1
- KEY_F2
- KEY_F3
- KEY_F4
- KEY_F5
- KEY_F6
- KEY_F7
- KEY_F8
- KEY_F9
- KEY_F10
- KEY_F11
- KEY_F12

Estados das teclas:

- KEY_STATE_DOWN
- KEY_STATE_UP

Tratando Cliques do Mouse

- Para poder tratar os eventos gerados pelo mouse (**cliques do mouse**) é necessário criar uma função para essa tarefa.

- Essa função deve ter a seguinte sintaxe:

```
void MouseClickInput(int button, int state, int x, int y)
{
   /* Bloco de Comandos */
}
```

- Também é **necessário indicar** que essa é a sua função para tratar eventos de clique do mouse:

```
graphics.SetMouseClickInput(MouseClickInput);
```

Tratando Cliques do Mouse

- Dessa forma, sempre que um botão do mouse for pressionado a função **MouseClickInput** será executada e o parâmetro button indicará qual botão foi pressionado. Os parâmetros x e y indicam a posição na tela em que mouse estava quando o clique foi realizado.

- **Exemplo:**

```
void MouseClickInput(int button, int state, int x, int y)
{
  if ((button == MOUSE_LEFT_BUTTON)&&(state == MOUSE_STATE_DOWN))
  {
     destino_x = x;
     destino_y = y;
  }
}
```

Se o botão esquerdo foi pressionado

As variáveis destino_x e destino_y recebem a posição x e y do mouse no momento do clique, ou seja, onde o usuário clicou.

visionlab
visualization laboratory

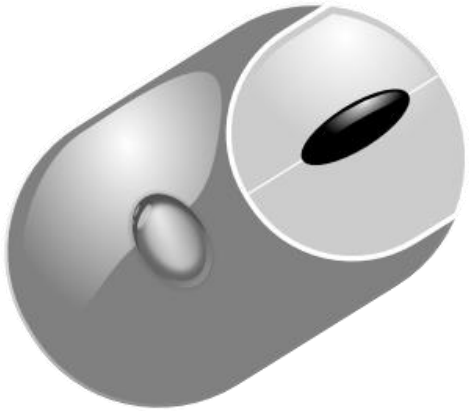
Códigos da Teclas do Mouse

Botões:

- MOUSE_LEFT_BUTTON
- MOUSE_MIDDLE_BUTTON
- MOUSE_RIGHT_BUTTON

Estados:

- MOUSE_STATE_DOWN
- MOUSE_STATE_UP



Tratando o Movimento do Mouse

- Para poder tratar os eventos de movimento gerados pelo mouse é necessário criar uma função para essa tarefa.

- Essa função deve ter a seguinte sintaxe:

```
void MouseMotionInput(int x, int y)
{
   /* Bloco de Comandos */
}
```

- Também é **necessário indicar** que essa é a sua função para tratar eventos de movimento do mouse:

```
graphics.SetMouseMotionInput(MouseMotionInput);
```

Tratando o Movimento do Mouse

- Dessa forma, sempre que o mouse for movimentado pelo usuário a função **MouseClickInput** será executada e os parâmetros x e y indicaram a posição do mouse na tela.

- **Exemplo:**

```
void MouseMotionInput(int x, int y)
{
  mouse_x = x;
  mouse_y = y;
}
```

As variáveis mouse_x e mouse_y recebem a posição x e y do mouse, ou seja, o local onde o usuário está com o cursor do mouse.

Tratando Cliques do Mouse Sobre uma Imagem

- Para poder tratar os eventos de clique do mouse sobre uma determinada imagem é necessário definir uma função para essa tarefa.

- A função para tratar esse evento deve ter a seguinte sintaxe:

```
void MouseClickMinhaImagem(int button, int state, int x, int y)
{
   /* Bloco de Comandos */
}
```

- Também é necessário indicar que essa é a sua função para tratar eventos de clique do mouse sobre a imagem em questão usando o comando SetOnClick:

```
MinhaImagem.SetOnClick(MouseClickMinhaImagem);
```

Tratando Cliques do Mouse Sobre uma Imagem

- Dessa forma, sempre que o usuário clicar sobre a imagem “MinhaImagem”, a função MouseClickMinhaImagem será executada e o parâmetro button indicará qual botão foi pressionado. Os parâmetros x e y indicam a posição na tela relativa a imagem em que mouse estava quando o clique foi realizado.

- **Exemplo:**

```
void MouseClickMinhaImagem(int button, int state, int x, int y)
{
  carregando_imagem = true;
}
```

Tratando Cliques do Mouse Sobre uma Imagem

- **Importante:** Para poder usar este evento é necessário que a posição da imagem tenha sido definida com o comando SetPosition.

- **Exemplo:**

```
Image minha_imagem;

void MouseClickMinhaImagem(int button, int state, int x, int y)
{
  clicou_na_imagem = true;
}

int main(void)
{
  ...
  minha_imagem.LoadPNGImage("Marvin.png");
  minha_imagem.SetPosition(0,100,256,256);
  minha_imagem.SetOnClick(MouseClickMarvin);
  ...
}
```

Exercícios

Lista de Exercícios 07 – Imagens e Interação

http://www.inf.puc-rio.br/~elima/prog-jogos/