Introdução a Prática em OpenGL

Universidade de São Paulo – USP
Disciplina de Computação Gráfica
Profª Maria Cristina
PAE: Thiago Silva Reis Santos
Agosto de 2010

Sumário

- Bibliotecas Necessárias
- Instalação do Dev-C++
- Instalando e Rodando - Ubuntu
- Primeiro Programa em OpenGL
- Padronização dos Nomes
- GLUT – GL Utility Toolkit
- GLUT – Principais Funções
- GLUT – Tratamento de Eventos
- Visualização Bidimensional
- OpenGL Básico
- Exercícios

Bibliotecas Necessárias

- Para executar aplicações OpenGL é necessário ter instalados as três DLL:
  - OpenGL.DLL (pré-instalada no windows)
  - GLU.DLL
  - GLUT.DLL

- Deve-se colocar esses arquivos na pasta system32 (no WindowsXP na pasta C:\WINDOWS\system32).

Instalação do Dev-C++

- Para desenvolvermos pequenos projetos em OpenGL faz-se necessário um ambiente de desenvolvimento (IDE) e o compilador.
- Neste curso usaremos a linguagem C++ e como IDE o Dev-C++. Acesse o site e siga os passos para a instalação da IDE.
- www.bloodshed.net/devcpp.html

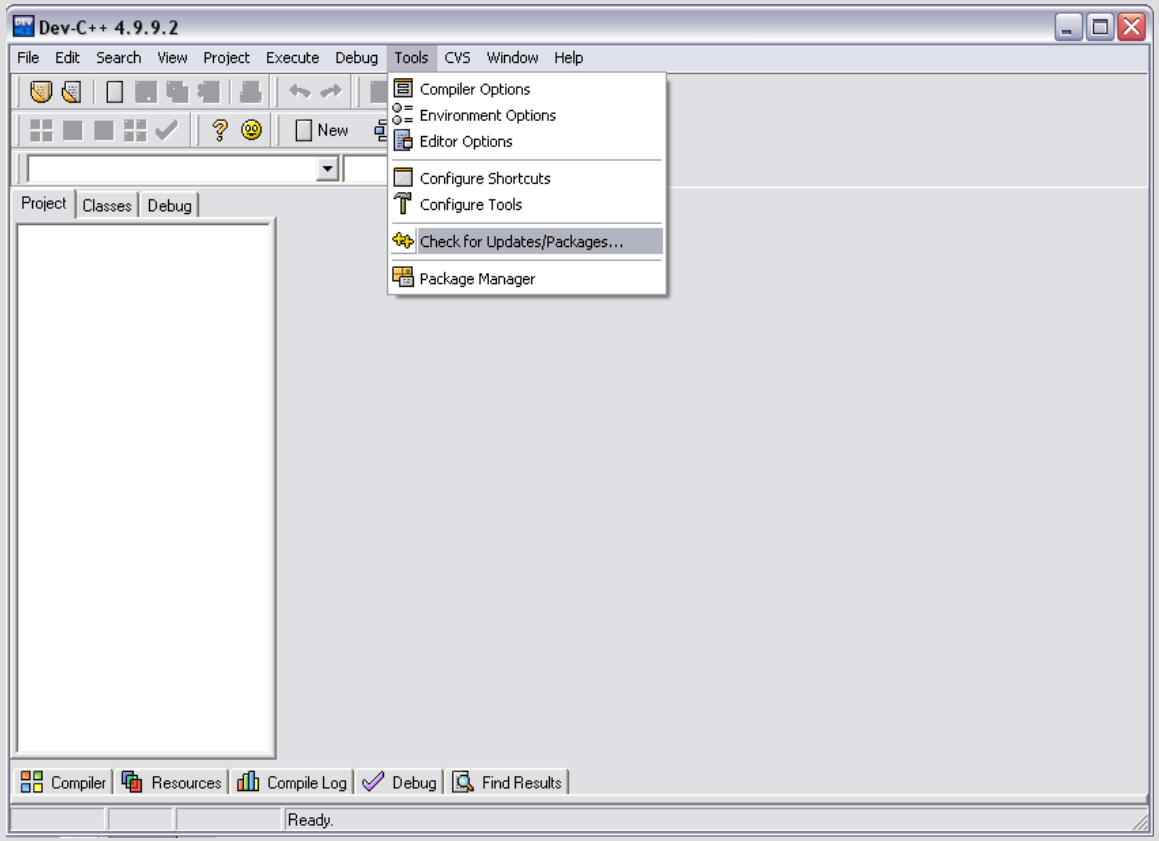
Instalação do Dev-C++

- O Dev-C++ já traz todo o suporte para o desenvolvimento de programas em OpenGL. Porém o Dev-C++ não vem com as bibliotecas GLUT (OpenGL Utility Toolkit – gerenciamento de interfaces).

- Devemos instalar a GLUT via Dev-C++, conforme os seguintes passos:

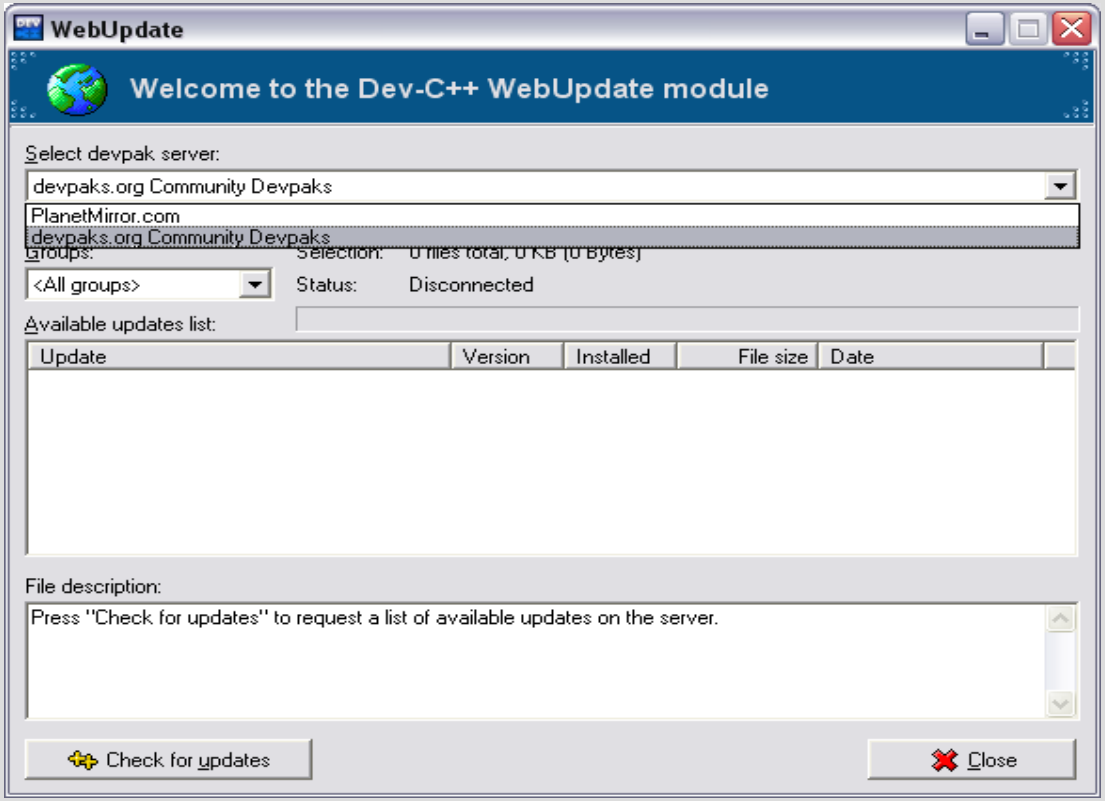
Instalação do Dev-C++

- Inicie o Dev-C++ e em seguida vá em : Ferramentas/Atualizações (Tools/Check for Updates)



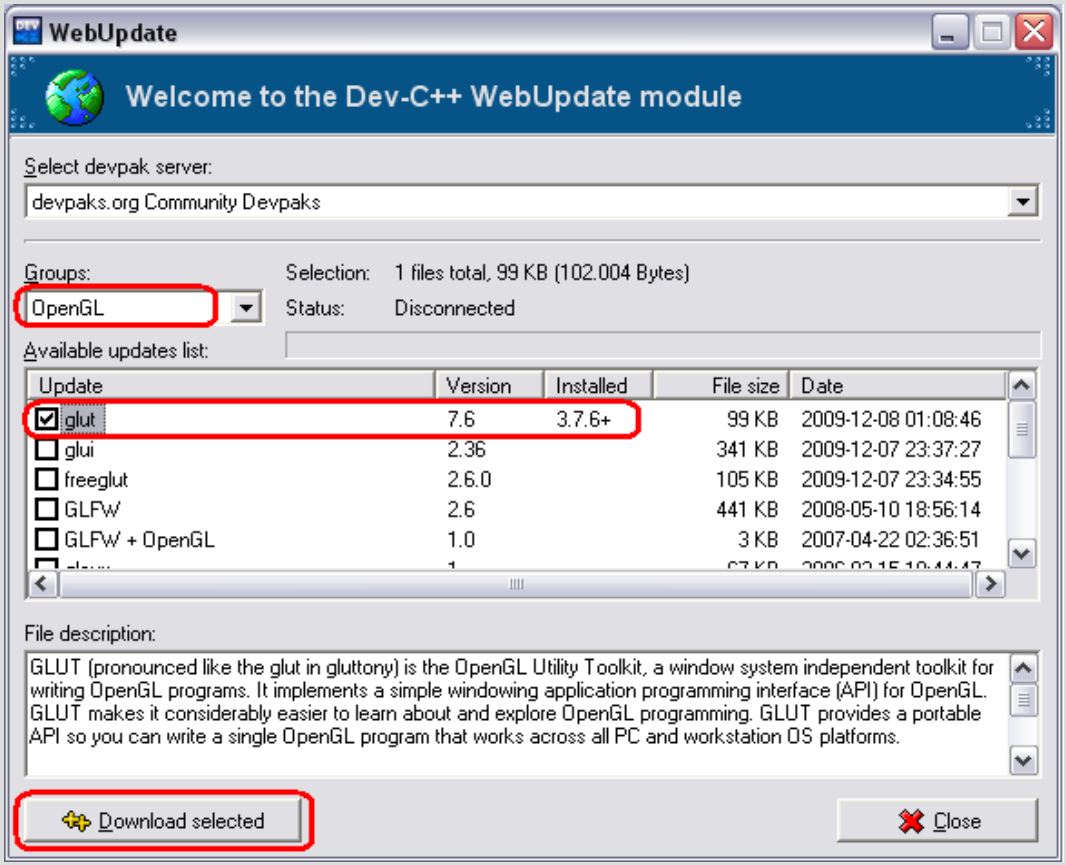
Instalação do Dev-C++

- Escolha a opção “devpaks.org Community Devpaks” e click em “Check for Updates”.



Instalação do Dev-C++

- Selecione o grupo “OpenGL”, depois click em “glut” e “Download Selected”.



Instalando e Rodando - Ubuntu

- Infelizmente não há (até agora) Dev-C++ para o Ubuntu
- Porém é possível programar e compilar usando um editor de texto qualquer.

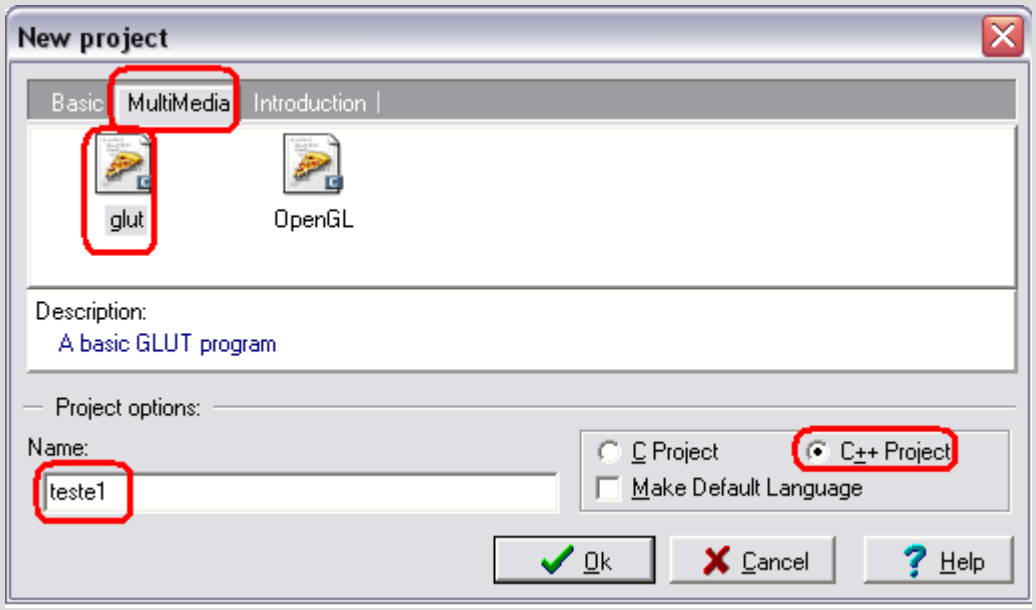
```
gcc programa.c -o programa -lglut
```

- Toda distribuição Linux já vem com o OpenGL instalado, no entanto, é necessário instalar a glut

```
sudo apt-get install lglut
```

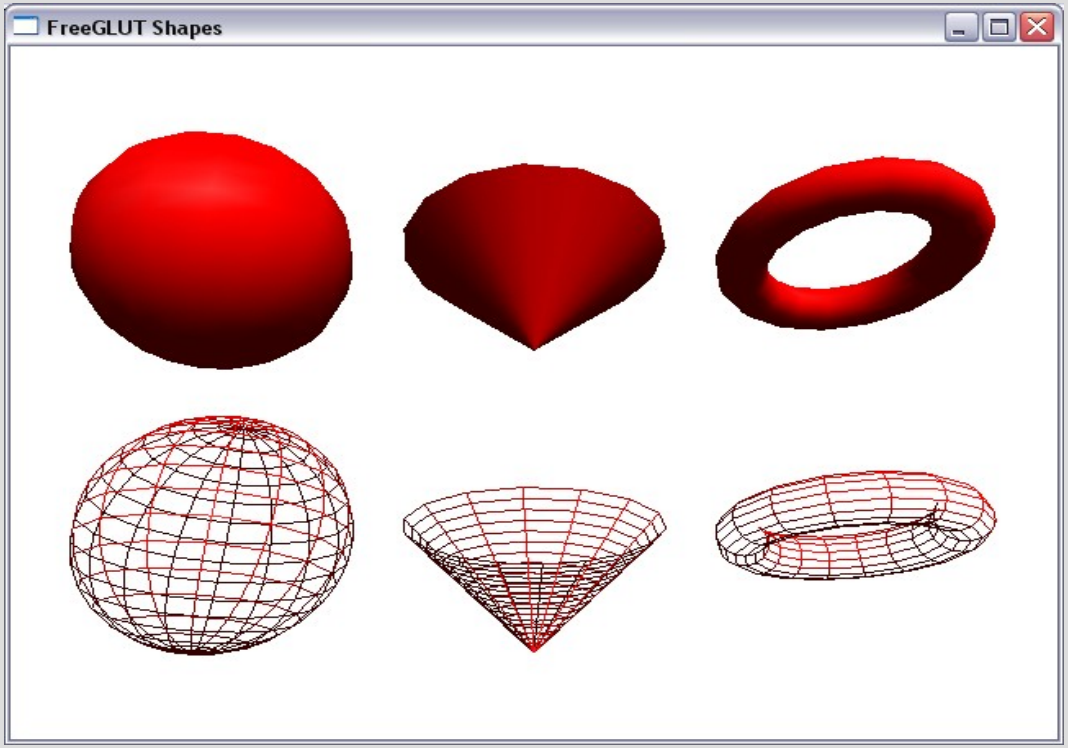
Primeiro Programa em OpenGL

- Inicie o Dev-C++ em seguida vá em “arquivo” e escolha a opção “Novo/projeto”. Na janela que aparece click na aba “MultiMedia” e selecione o ícone “glut” e “projeto C++”. Adicione o nome ex.: teste1 e click “Ok”.



Primeiro Programa em OpenGL

- O Dev-C++ gera um programa exemplo em OpenGL no arquivo main.cpp que podemos compilar e executar com F9.



Padronização dos Nomes

- Para padronizar e facilitar sua utilização, as funções em OpenGL segue a seguinte convenção:
  - <PrefixoBiblioteca>
  - <ComandoRaiz>
  - <ContadorArgumentosOpcional>
  - <TipoArgumentosOpcional>
  - Ex.: void glColor3f(GLfloat red, Glfloat green, Glfloat blue)

Padronização dos Nomes

- Com esta padronização dos nomes, existirão várias funções que executam a mesma ação porém com nomes distintos, variando o número de argumentos.
  - void glColor3i(GLint red, …);
  - void glColor3d(GLdouble red, …);
  - void glColor4f(GLfloat red, ..., Glfloat alpha);

Padronização dos Nomes

- Valores possíveis para o tipo de argumento

Argumento	Descrição
b	Para argumento do tipo signed char
s	Para argumento do tipo short
i	Para argumento do tipo interger
f	Para argumento do tipo float
d	Para argumento do tipo double
ub	Para argumento do tipo unsigned char
us	Para argumento do tipo unsigned short
ui	Para argumento do tipo unsigned integer
*v	Para ser qualquer um dos tipos anteriores, e v indica que é um argumento do tipo ponteiro para um vetor especificado no lugar de * (ex.: fv é um vetor de floats, iv é um vetor de inteiros)

Padronização dos Nomes

- Tipos de dados do OpenGL

Tipo OpenGL	Representação	Tipo C	Sufixo
GLbyte	Inteiro de 8 bits	signed char	b
GLshort	Inteiro de 16 bits	short	s
GLint, GLsizei	Inteiro de 32 bits	int ou long (depende)	i
GLfloat, GLclampf	Ponto flutuante 32 bits	float	f
GLdouble, GLclampd	Ponto flutuante 64 bits	double	d
GLubyte, GLboolean	Inteiro de 8 bits sem sinal	unsigned char	ub
GLushort	Inteiro de 16 bits sem sinal	unsigned short	us
GLuint, GLenum	Inteiro de 32 bits sem sinal	unsigned int	ui

GLUT – GL Utility Toolkit

- A OpenGL trata da exibição de objetos gráficos. Não possuindo nenhum código para o tratamento de eventos de mouse, gerenciamento de janelas, etc.
- A GLUT é uma biblioteca portável responsável pelo gerenciamento de janelas; eventos de mouse, teclado e joystick; criação de menus e ativar os desenhos de objetos OpenGL.

GLUT – Principais Funções

- void glutInit(): função que inicia a GLUT
- glutInitDisplayMode: define o modo da GLUT
  - GLUT_DOUBLE; GLUT_SINGLE; GLUT_DEPTH
- void glutInitWindowPosition(int x, int y)
  - Posiciona a janela nas coordenas x,y
- void glutInitWindowSize(int w, int h)
  - Define a largura (w) e a altura (h)

GLUT – Principais Funções

- int glutCreateWindow(char *name)
  - Cria a janela com o nome definido
- void glut DestroyWindow(int win)
  - Destrói uma janela GLUT

GLUT – Tratamento de Eventos

- O gerenciamento dos eventos ocorre através das funções de *callback*.

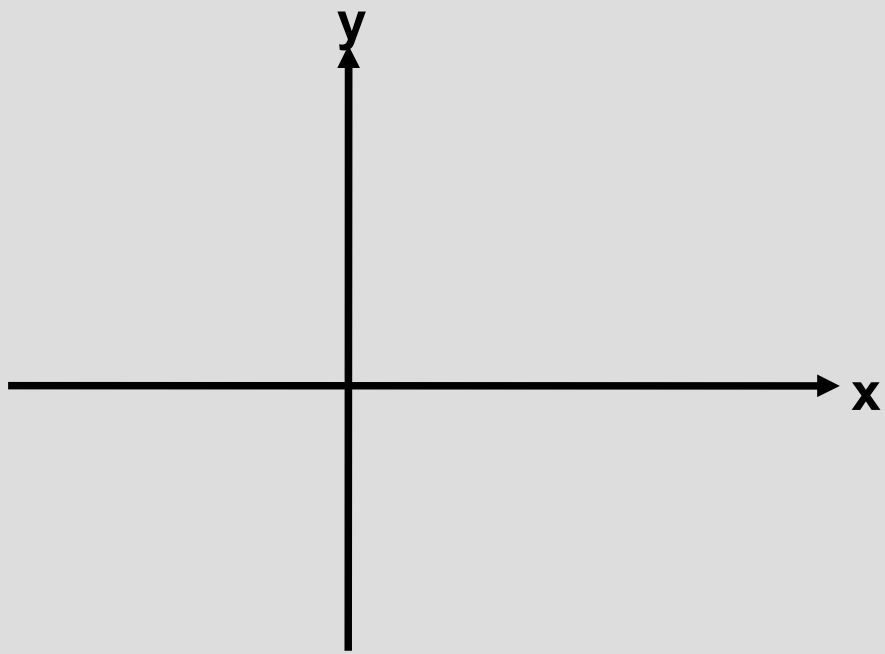
- void glutDisplayFunc
  - Responsável por desenhar a janela
- void glutReshapeFunc
  - Responsável pelo redimensionamento da janela
- void glutKeyboardFunc
  - Responsável pelos eventos de teclado

GLUT – Tratamento de Eventos

- void glutMouseFunc
  - Responsável pelos eventos de mouse
- void MotionFunc
  - Responsável pelos movimentos do mouse
- void glutMainLoop
  - GLUT inicia o gerenciamento dos eventos

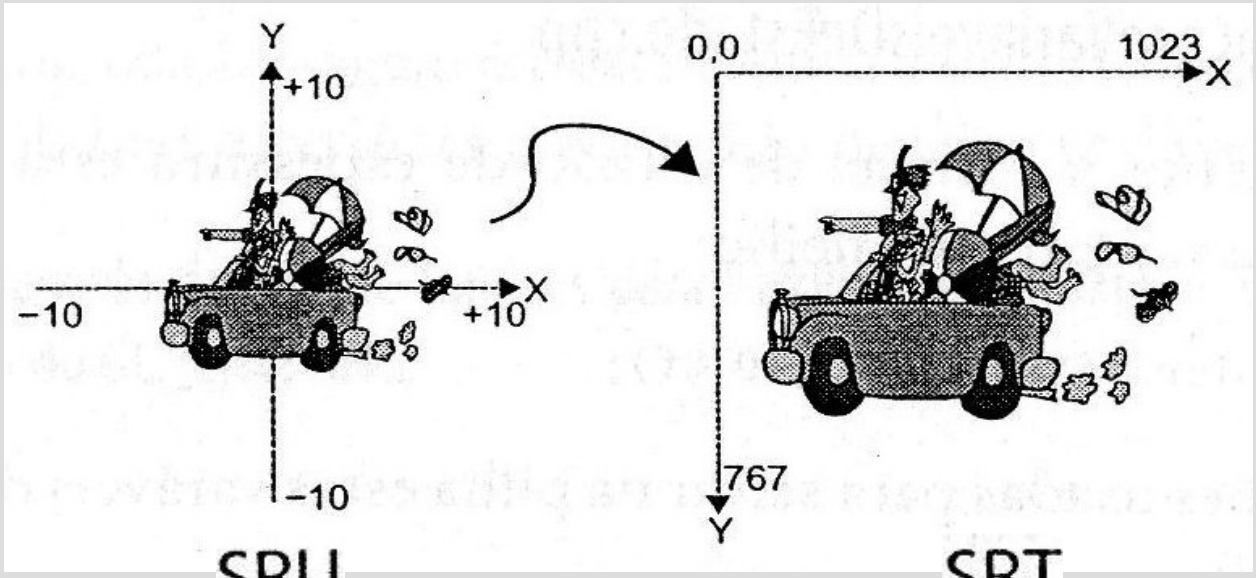
Visualização Bidimensional

- Todos os nossos modelos são projetados em um plano cartesiano infinito chamado sistema de referência do universo (SRU).



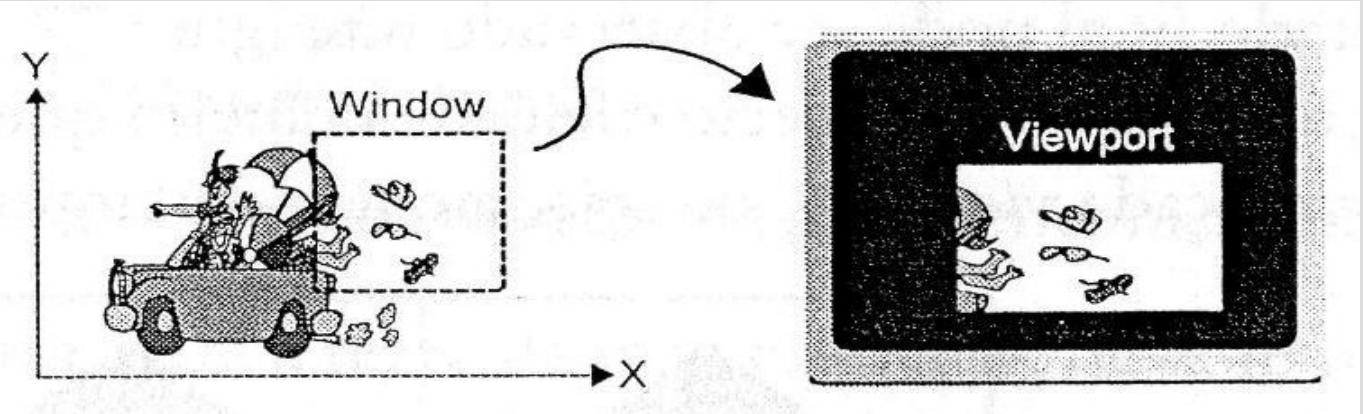
Visualização Bidimensional

- O monitor do computador possui o sistema de referência da tela (SRT), cuja a origem é o canto superior esquerdo. O SRT é limitado e depende de cada monitor. Faz-se necessário um mapeamento entre SRU e SRT.



Visualização Bidimensional

- Para realizar o mapeamento entre SRU e SRT é preciso definir a região de interesse do usuário no “Universo”. Essa região chama-se *window* (janela de seleção).
- Apenas a *window* será mapeada no SRT.
- A região do SRT que receberá a *window* é chamada de *viewport*.



Visualização Bidimensional

- Em OpenGL nós definimos a windows e o viewport com as seguintes funções:
  - void glutOrtho2D(GLdouble esq, GLdouble dir, GLdouble baixo, GLdouble cima);
    - Função que define a *window* no SRU. Com os valores de x a esquerda e direita e os valores de y em baixo e em cima.
  - void glViewport(GLint x, GLint y, GLint larg, GLint alt)
    - Função que define a viewport no SRT. Os dois primeiros valores (x,y) são o canto inferior esquerdo seguido da largura e altura.

OpenGL Básico

- Para garantir que as funções OpenGL e GLUT sejam reconhecidas.
  - #include <GL/glut.h>

- Para limpar a tela usa-se duas funções, uma define a cor de fundo a outra pinta o fundo com a cor definida.
  - glClearColor(1,1,1,0);
  - glClear(GL_COLOR_BUFFER_BIT);

OpenGL Básico

- Para desenhar um triangulo com cor em cada vértice.

  - glBegin(GL_TRIANGLE);
    - glColor3f(1,0,0);
    - glVertex2f(-1, 0);
    - glColor3f(0,1,0);
    - glVertex2f(1, 0);
    - glColor3f(0,0,1);
    - glVertex2f(0, 1);
  - glEnd();

OpenGL Básico

- Para tratar os eventos de teclado usa-se a codificação ASCII. O exemplo abaixo verifica se a tecla ESC foi pressionada.
  - if (key == 27)
    - exit(0);

27	1B	Escape	ESC
28	1C	File separator	FS
29	1D	Group separator	GS
30	1E	Record separator	RS
31	1F	Unit separator	US

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
32	20	Space	64	40	@	96	60	`
33	21	!	65	41	A	97	61	a
34	22	"	66	42	B	98	62	b
35	23	#	67	43	C	99	63	c
36	24	$	68	44	D	100	64	d
37	25	%	69	45	E	101	65	e
38	26	&	70	46	F	102	66	f
39	27	'	71	47	G	103	67	g
40	28	(	72	48	H	104	68	h
41	29	)	73	49	I	105	69	i
42	2A	*	74	4A	J	106	6A	j
43	2B	+	75	4B	K	107	6B	k
44	2C	,	76	4C	L	108	6C	l
45	2D	-	77	4D	M	109	6D	m
46	2E	.	78	4E	N	110	6E	n
47	2F	/	79	4F	O	111	6F	o
48	30	0	80	50	P	112	70	p
49	31	1	81	51	Q	113	71	q
50	32	2	82	52	R	114	72	r
51	33	3	83	53	S	115	73	s
52	34	4	84	54	T	116	74	t
53	35	5	85	55	U	117	75	u
54	36	6	86	56	V	118	76	v
55	37	7	87	57	W	119	77	w
56	38	8	88	58	X	120	78	x
57	39	9	89	59	Y	121	79	y
58	3A	:	90	5A	Z	122	7A	z
59	3B	;	91	5B	[	123	7B	{
60	3C	<	92	5C	\	124	7C	|
61	3D	=	93	5D	]	125	7D	}
62	3E	>	94	5E	^	126	7E	~
63	3F	?	95	5F	_	127	7F	DEL

Exercício

- 1º) Veja o seguinte programa que apresenta um triangulo. Faça as seguintes alterações:
  - a) Quando o usuário pressionar as teclas 1 a 9 o programa deve gerar o número solicitado de triângulos.
  - b) Agora além de gerar o número solicitado de triângulos você deve gerar ou triângulo (GL_TRIANGLES), se pressionado t, ou quadrados (GL_QUADS) se pressionado q.

Pratica1.zip

Exercício

- 2º) Desenhe uma casa com as primitivas de triângulos e quadrados. Depois faça as seguintes alterações:
  - a) Pinte a casa com alguma cor.
  - b) Quando pressionado um botão no mouse as cores devem mudar.

Bibliografia

- COHEN, Marcelo; MANSSOUR, Isabel H. OpenGL: Uma Abordagem Prática e Objetiva. São Paulo, Editora: Novatec. 2006.