GLUT

Interface de programmation

OpenGL Utility Toolkit

Table des matières

1 Initialisation 6

1.1 glutInit 6

1.2 glutInitWindowSize, glutInitWindowPosition 6

1.3 glutInitDisplayMode 7

2 Traitement des événements 8

2.1 glutMainLoop 8

3 Gestion de fenêtre 8

3.1 GlutCreateWindow 8

3.2 glutCreateSubWindow 9

3.3 glutSetWindow 9

3.4 glutGetWindow 9

3.5 glutDestroyWindow 10

3.6 glutPostRedisplay 10

3.7 glutSwapBuffers 10

3.8 glutPositionWindow 11

3.9 glutReshapeWindow 11

3.10 glutFullScreen 12

3.11 glutPopWindow, glutPushWindow 12

3.12 glutShowWindow, glutHideWindow, glutIconifyWindow 12

3.13 glutSetWindowTitle, glutSetIconTitle (char *name); 13

3.14 glutSetCursor 13

4 Gestion de plans de superposition (recouvrement) 14

4.1 glutEstablishOverlay 14

4.2 glutUseLayer 14

4.3 glutRemoveOverlay 15

4.4 glutPostOverlayRedisplay 15

4.5 glutShowOverlay et glutHideOverlay 16

5 Gestion de menus 16

5.1 glutCreateMenu 16

5.2 glutSetMenu 17

5.3 glutDestroyMenu 17

5.4 glutAddMenuEntry 17

5.5 glutAddSubMenu 17

5.6 glutChangeToMenuEntry 18

5.7 glutChangeToSubMenu 18

5.8 glutRemoveMenuItem 19

5.9 glutAttachMenu, glutDetachMenu 19

6 Inscription des fonctions de rappel 19

6.1 glutDisplayFunc 20

6.2 glutOverlayDisplayFunc 21

6.3 glutReshapeFunc 21

6.4 glutKeyboardFunc 22

6.5 glutMouseFunc 23

6.6 glutMotionFunc, glutPassiveMotionFunc 23

6.7 glutVisibilityFunc 24

6.8 glutEntryFunc 24

6.9 glutSpecialFunc 25

6.10 glutSpaceballMotionFunc 26

6.11 glutSpaceballRotateFunc 26

6.12 glutSpaceballButtonFunc 26

6.13 glutButtonBoxFunc 26

6.14 glutDialsFunc 27

6.15 glutTabletMotionFunc 27
6.16 glutTabletButtonFunc 27
6.17 glutMenuStatusFunc, glutMenuStateFunc 27
6.18 glutIdleFunc 28
6.19 glutTimerFunc 29

7 Gestion des index de couleur 29

7.1 glutSetColor 29
7.2 glutGetColor 30
7.3 glutCopyColormap 30

8 Obtenir l'état 31

8.1 glutGet 31
8.2 glutLayerGet 33
8.3 glutDeviceGet 33
8.4 glutGetModifiers 34
8.5 glutExtensionsSupported 34

9 Rendu des polices de caractères 35

9.1 glutBitmapCharacter 35
9.2 glutBitmapWidth 36
9.3 glutStrokeCharacter 36
9.4 glutStrokeWidth 36

10 Rendu d'objets géométriques 37

10.1 glutSolidSphere, glutWireSphere 37
10.2 glutSolidCube, glutWireCube 37
10.3 glutSolidCone, glutWireCone 37
10.4 glutSolidTorus, glutWireTorus 38
10.5 glutSolidDodecahedron, glutWireDodecahedron 38
10.6 glutSolidOctahedron, glutWireOctahedron 38

10.7 glutSolidTetrahedron, glutWireTetrahedron..... 39

10.8 glutSolidIcosahedron, glutWireIcosahedron 39

10.9 glutSolidTeapot, glutWireTeapot 39

1 Initialisation

1.1 glutInit

La fonction glutInit initialise la bibliothèque GLUT.

1.1.1 Usage

void glutInit (int *argcp, char **argv);

argcp	Un pointeur à la variable non modifiée *argcp* du programme *main*. Au retour, *argcp* est modifiée parce que glutInit extrait les options de la ligne de commande.
argv	Un pointeur à la variable non modifiée *argv* du programme *main*. *argv* est mis à jour parce que les options de la ligne de commande utilisée par GLUT sont extraites.

1.1.2 Description

La fonction glutInit initialise la bibliothèque GLUT et négocie une session avec le système de fenêtrage. Elle traite également les lignes de commandes qui sont propres à chaque système de fenêtrage. Si une erreur survient, la fonction peut se terminer.

1.1.3 Paramètres pour le système X

Les paramètres de la ligne de commande qui sont compris par la bibliothèque GLUT sont:

-display DISPLAY	Spécifie l'adresse du serveur X auquel se connecter. Si ce n'est spécifié, la variable d'environnement est utilisée.
-geometry WxH+X+Y	Détermine la position de la fenêtre sur l'écran. Le paramètre de geometry doit être formaté selon la spécification standard de X.
-iconic	Indique que la fenêtre est créée dans l'état iconisé.
-indirect	Force l'utilisation du contexte indirect de rendu réaliste d'OpenGL.
-direct	Force l'utilisation du contexte direct de rendu réaliste d'OpenGL. Une erreur fatale est rencontrée si le système ne supporte pas ce mode. Par défaut, le mode direct est utilisé, sinon c'est le mode indirect.
-gldebug	Après le traitement des fonctions de rappel ou des événements, vérifier s'il y a des erreurs d'OpenGL en appelant glGetError. S'il y a une erreur, imprimer un avertissement obtenu par la fonction gluErrorString.
-sync	Utiliser le protocole X synchronisé. Plus facile pour retracer les erreurs potentielles du protocole X.

1.2 glutInitWindowSize, glutInitWindowPosition

Les fonctions glutInitWindowSize et glutInitWindowPosition permettent de positionner la fenêtre sur l'écran et d'en spécifier la taille.

1.2.1 Usage

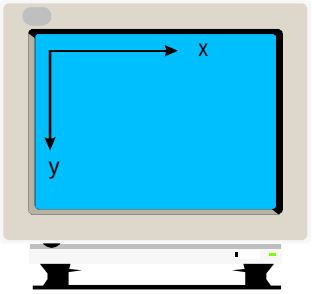
void glutInitWindowSize (int width, int height);

void glutInitWindowPosition (int x, int y);

width	Largeur de la fenêtre en pixels.
height	Hauteur de la fenêtre en pixels.
x	Position en *x* du coin gauche supérieur de la fenêtre.
y	Position en *y* du coin gauche supérieur de la fenêtre.

1.2.2 Description

Ces fonctions indiquent qu'une fenêtre sera créée (avec la fonction glutCreateWindow) à la position initiale spécifiée par la fonction glutInitWindowPosition et de la taille spécifiée par la fonction glutInitWindowSize. La valeur par défaut est (-1, -1) qui correspond au coin supérieur gauche du moniteur et la taille initiale est de 300x300 pixels. Cependant, le système de fenêtrage n'est pas forcé de se conformer exactement à la position spécifiée et à la taille demandée. Une application utilisant la bibliothèque GLUT devrait utiliser la fonction de rappel pour redimensionner la fenêtre pour déterminer la vraie taille d'une fenêtre.



1.3 glutInitDisplayMode

Cette fonction spécifie le mode d'affichage de la fenêtre.

1.3.1 Usage

void glutInitDisplayMode (unsigned int *mode*);

mode Mode d'affichage qui est en général une opération *or* bit à bit de masque de bits. Les valeurs permises sont :

GLUT_RGBA	Masque de bits pour choisir une fenêtre en mode RGBA. C'est la valeur par défaut si GLUT_RGBA ou GLUT_INDEX ne sont spécifiés.
GLUT_RGB	Un alias de GLUT_RGBA.
GLUT_INDEX	Masque de bits pour choisir une fenêtre en mode index de couleur. Ceci l'emporte si GLUT_RGBA est spécifié.
GLUT_SINGLE	Masque de bits pour spécifier un tampon simple pour la fenêtre. Ceci est la valeur par défaut.
GLUT_DOUBLE	Masque de bit pour spécifier une fenêtre avec un double tampon. Cette valeur l'emporte sur GLUT_SINGLE.
GLUT_ACCUM	Masque de bits pour choisir une fenêtre avec un tampon d'accumulation.
GLUT_RGBA	Masque de bits pour choisir une fenêtre avec une composante *alpha* pour le tampon de couleur.
GLUT_DEPTH	Masque de bits pour choisir une fenêtre avec un tampon de profondeur.
GLUT_STENCIL	Masque de bits pour choisir une fenêtre avec un tampon de pochoir.
GLUT_MULTISAMPLE	
GLUT_STEREO	Masque de bits pour choisir une fenêtre stéréo.

1.3.2 Description

Le mode d'affichage est utilisé pour créer les fenêtres et les sous-fenêtres. Le mode GLUT_RGBA permet d'obtenir une fenêtre utilisant le modèle de couleur RGB mais si on désire utiliser dans ce modèle la composante alpha, on doit utiliser le mode d'affichage GLUT_RGBA.

2 Traitement des événements

2.1 glutMainLoop

Cette fonction permet d'entrer dans la boucle de GLUT de traitement des événements.

2.1.1 Usage

```
void glutMainLoop ( void );
```

2.1.2 Description

Cette fonction permet d'entrer dans la boucle de GLUT de traitement des événements. Cette fonction est appelée seulement une fois dans une application. Dans cette boucle, les fonctions de rappel qui ont été enregistrées sont appelées à tour de rôle.

3 Gestion de fenêtre

3.1 GlutCreateWindow

Cette fonction crée une fenêtre en utilisant le système de fenêtrage du système.

3.1.1 Usage

```
int glutCreateWindow ( char * name );
```

name Chaîne de caractères identifiant la fenêtre.

3.1.2 Description

Cette fonction crée une fenêtre. Le nom de la fenêtre dans la barre de titre de la fenêtre prend la valeur de la chaîne de caractères spécifiée par name. Cette fonction retourne un entier positif identifiant le numéro de la fenêtre. Cet entier peut par la suite être utilisé par la fonction glutSetWindow.

Chaque fenêtre possède un contexte unique d'OpenGL. Un changement d'état de la fenêtre associée au contexte d'OpenGL peut être effectué une fois la fenêtre créée. L'état d'affichage de la fenêtre à afficher n'est pas actualisé tant que l'application n'est pas entrée dans la fonction glutMainLoop. Ce qui signifie qu'aucun objet graphique ne peut être affiché dans la fenêtre, parce que la fenêtre n'est pas encore affichée.

3.2 glutCreateSubWindow

La fonction glutCreateSubWindow crée une sous-fenêtre à l'intérieur d'une fenêtre parent.

3.2.1 Usage

```
int   glutCreateSubWindow ( int win, int x, int y, int width, int height );
```

win	Identificateur de la fenêtre parent.
x	Coordonnées en pixels de la position en *x* relativement à la fenêtre parent.
y	Coordonnées en pixels de la position en *y* relativement à la fenêtre parent.
width	Largeur en pixels.
height	Hauteur en pixels.

3.2.2 Description

Cette fonction crée une sous-fenêtre dans la fenêtre parent *win*, de largeur *width*, et de hauteur *height*. La valeur retournée par cette fonction est un entier positif identifiant la sous-fenêtre. On peut créer une sous-fenêtre à l'intérieur d'un autre sous-fenêtre parent.

L'état d'affichage de la fenêtre à afficher n'est pas actualisé tant que l'application n'est pas entrée dans la fonction glutMainLoop. Ce qui signifie qu'aucun objet graphique ne peut être affiché dans la fenêtre, parce que la fenêtre n'est pas encore affichée.

Il est à remarquer que l'origine se trouve dans le coin supérieur gauche de la fenêtre.

3.3 glutSetWindow

Cette fonction établit quelle est la fenêtre courante.

3.3.1 Usage

```
void glutSetWindow ( int win );
```

win	Identificateur de la fenêtre courante.

3.3.2 Description

Cette fonction établit que la fenêtre identifiée par *win* devient la *fenêtre courante*.

3.4 glutGetWindow

Cette fonction retourne le numéro de la *fenêtre courante*.

3.4.1 Usage

```
int glutGetWindow ( void );
```

3.4.2 Description

Cette fonction retourne le numéro de la fenêtre courante. Si la fenêtre courante a été détruite, alors le numéro retourné est 0.

3.5 glutDestroyWindow

Cette fonction détruit une fenêtre.

3.5.1 Usage

```
void glutDestroyWindow ( int win );
```

Win Identificateur de la fenêtre à détruire.

3.5.2 Description

glutDestroyWindow détruit la fenêtre identifiée par le paramètre *win*. Elle détruit également le contexte OpenGL associée à la fenêtre, la table des index de couleurs (s'il y lieu), le plan de recouvrement (s'il y a lieu) et son état. Si *win* identifie la fenêtre courante, alors la fenêtre courante devient invalide (glutGetWindow retourne la valeur 0).

3.6 glutPostRedisplay

glutPostRedisplay indique que la fenêtre doit être rafraîchie.

3.6.1 Usage

```
void glutPostRedisplay ( void );
```

3.6.2 Description

Cette fonction indique que le plan normal de la *fenêtre courante* doit être réaffiché. Lors de la prochaine itération dans la boucle principale de glutMainLoop, la fonction de rappel d'affichage est appelée et le plan normal est affiché. Plusieurs appels à la fonction glutPostRedisplay n'engendrent qu'un seul rafraîchissement.

Logiquement, une fenêtre endommagée est marquée comme devant être rafraîchie, ce qui est équivalent à faire appel la fonction glutPostRedisplay.

3.7 glutSwapBuffers

glutSwapBuffers échange les tampons de la fenêtre courante si le mode double tamponnage est activé.

3.7.1 Usage

```
void glutSwapBuffers ( void );
```

3.7.2 Description

Cette fonction échange les tampons de la couche en utilisation de la fenêtre courante. En fait, le contenu du tampon arrière de la couche en utilisation de la fenêtre courante devient le contenu du tampon avant. Le contenu du tampon arrière devient indéfini.

La fonction glFlush est appelée implicitement par glutSwapBuffers. On peut exécuter des commandes d'OpenGL immédiatement après glutSwapBuffers, mais elles prennent effet lorsque

l'échange de tampon est complété. Si le mode double tamponnage n'est pas activé, cette fonction n'a aucun effet.

3.8 glutPositionWindow

glutPositionWindow demande un changement de la position de la fenêtre courante.

3.8.1 Usage

void glutPositionWindow (int x, int y);

x	Coordonnée en *x* de la position en pixels de la fenêtre.
y	Coordonnée en *y* de la position en pixels de la fenêtre.

3.8.2 Description

glutPositionWindow demande un changement de position de la fenêtre courante. Les coordonnées *x* et *y* sont des décalages par rapport à l'origine de l'écran pour les fenêtres de base. Pour les sous-fenêtres, il s'agit de décalage par rapport à l'origine de la fenêtre parent.

Le changement de position n'est pas immédiatement effectué, mais le changement est effectué lorsque l'application retourne dans la boucle principale. On peut donc grouper des appels aux fonctions glutPositionWindow, glutReshapeWindow et glutFullScreen.

Pour une fenêtre de base, le système de fenêtrage est libre d'appliquer face à la requête sa propre politique pour le positionnement de la fenêtre.

glutPositionWindow désactive le mode plein écran s'il est activé.

3.9 glutReshapeWindow

glutReshapeWindow demande un changement de dimensions de la fenêtre courante.

3.9.1 Usage

void glutReshapeWindow (int width, int height);

width	Nouvelle largeur en pixels de la fenêtre.
height	Nouvelle hauteur en pixels de la fenêtre.

3.9.2 Description

glutReshapeWindow demande un changement de dimensions de la fenêtre courante. Les paramètres *width* et *height* sont les nouvelles dimensions de la fenêtre et doivent être des entiers positifs.

Le changement de dimension n'est pas immédiatement effectué, mais le changement est effectué lorsque l'application retourne à la boucle principale. On peut donc grouper des appels aux fonctions glutPositionWindow, glutReshapeWindow et glutFullScreen.

Pour une fenêtre de base, le système de fenêtrage est libre d'appliquer face à la requête sa propre politique pour le dimensionnement de la fenêtre.

glutReshapeWindow désactive le mode plein écran s'il est activé.

3.10 glutFullScreen

glutFullScreen demande que la fenêtre courante soit en plein écran.

3.10.1 Usage

```
void glutFullScreen ( void );
```

3.10.2 Description

glutFullScreen demande que la fenêtre courante soit en plein écran. La sémantique de plein écran peut varier d'un système de fenêtrage à l'autre. Le but est d'obtenir la fenêtre la plus grande possible en la libérant des bordures et des barres de titre. Les dimensions de la fenêtre ne correspondent pas nécessairement aux dimensions de l'écran.

Le changement de dimension n'est pas immédiatement effectué, mais le changement est effectué lorsque l'application retourne à la boucle principale. On peut donc grouper des appels aux fonctions glutPositionWindow, glutReshapeWindow et glutFullScreen.

Par la suite, des appels aux fonctions glutReshapeWindow et glutPositionWindow désactivent le mode plein écran.

3.11 glutPopWindow, glutPushWindow

Ces fonctions changent l'ordre de la fenêtre courante par rapport à ses descendants.

3.11.1 Usage

```
void glutPopWindow ( void );

void glutPushWindow ( void );
```

3.11.2 Description

Les fonctions glutPopWindow et glutPushWindow s'appliquent aux fenêtres et aux sous-fenêtres. Elles permettent de changer l'ordre de la fenêtre courante. Ce changement n'est pas effectué immédiatement, mais le changement prend effet lorsque l'application retourne à la boucle principale.

3.12 glutShowWindow, glutHideWindow, glutIconifyWindow

3.12.1 Usage

```
void glutShowWindow ( void );

void glutHideWindow ( void );

void glutIconifyWindow ( void );
```

3.12.2 Description

La fonction glutShowWindow affiche la *fenêtre courante* (elle pourrait ne pas être visible si elle est occultée par une autre fenêtre. La fonction glutHideWindow cache la *fenêtre courante*. La fonction glutIconifyWindow met la *fenêtre courante* sous forme d'icône. Les actions de cacher, afficher ou icôniser une fenêtre ne sont pas effectués immédiatement. Les requêtes sont conservées pour exécution future lors du retour à la boucle principale des événements. Des requêtes additionnelles remplacent les requêtes emmagasinées. Les effets d'afficher, masquer ou icôniser une fenêtre dépendent de la politique d'affichage du système de fenêtrage.

3.13 glutSetWindowTitle, glutSetIconTitle (char *name);

3.13.1 Usage

```
void glutSetWindowTitle ( char *name );

void glutSetIconTitle ( char *name );
```

name Chaîne de caractères identifiant la fenêtre ou l'icône de la fenêtre.

3.13.2 Description

Ces fonctions s'appliquent à la *fenêtre courante* lorsque la *fenêtre courante* est une fenêtre de haut niveau ou fenêtre racine. Le nom d'une fenêtre ou d'une icône est établi lorsque de la création de la fenêtre par la fonction glutCreateWindow. Par la suite, le nom d'une fenêtre ou d'une icône peut être changé respectivement par des appels aux fonctions glutSetWindowTitle et glutSetIconTitle. Chaque appel à ces fonctions requiert que le système de fenêtrage change le titre de façon appropriée.

3.14 glutSetCursor

Cette fonction permet de changer l'apparence du réticule ou curseur.

3.14.1 Usage

```
void glutSetCursor ( int cursor );
```

cursor Identificateur du curseur.

GLUT_CURSOR_RIGHT_ARROW	Flèche pointant vers le haut et la droite.
GLUT_CURSOR_LEFT_ARROW	Flèche pointant vers le haut et la gauche.
GLUT_CURSOR_INFO	Main directionnelle.
GLUT_CURSOR_DESTROY	Crâne et os (tête de mort).
GLUT_CURSOR_HELP	Point d'interrogation.
GLUT_CURSOR_CYCLE	Flèche tournant en cercle.
GLUT_CURSOR_SPRAY	Aérosol.
GLUT_CURSOR_WAIT	Montre bracelet.
GLUT_CURSOR_TEXT	Point d'insertion pour le texte.
GLUT_CURSOR_CROSSHAIR	Croix.
GLUT_CURSOR_UP_DOWN	Curseur bidirectionnel pointant vers le haut et le bas.
GLUT_CURSOR_LEFT_RIGHT	Curseur bidirectionnel pointant vers la gauche et la droite.
GLUT_CURSOR_TOP_SIDE	Flèche pointant vers le côté supérieur.
GLUT_CURSOR_BOTTOM_SIDE	Flèche pointant vers le côté inférieur.
GLUT_CURSOR_LEFT_SIDE	Flèche pointant vers le côté gauche.

GLUT_CURSOR_RIGHT_SIDE	Flèche pointant vers le côté droit.
GLUT_CURSOR_TOP_LEFT_CORNER	Flèche pointant vers le coin supérieur gauche.
GLUT_CURSOR_TOP_RIGHT_CORNER	Flèche pointant vers le coin supérieur droit.
GLUT_CURSOR_BOTTOM_LEFT_CORNER	Flèche pointant vers le coin inférieur gauche.
GLUT_CURSOR_BOTTOM_RIGHT_CORNER	Flèche pointant vers le coin inférieur droit .
GLUT_CURSOR_FULL_CROSSHAIR	Grande croix.
GLUT_CURSOR_NONE	Curseur invisible.
GLUT_CURSOR_INHERIT	Utiliser le curseur parent.

3.14.2 Description

glutSetCursor change l'apparence du curseur pour la fenêtre courante. Chaque appel à cette fonction nécessite que le système de fenêtrage change l'apparence du réticule. L'apparence lorsqu'une fenêtre est créée est GLUT_CURSOR_INHERIT.

4 Gestion de plans de superposition (recouvrement)

Quand un plan de recouvrement existe matériel, la bibliothèque GLUT fournit un ensemble de fonctions pour établir, utiliser ou éliminer un plan de recouvrement pour une fenêtre de GLUT. Quand un plan de recouvrement est établit, un contexte séparé pour OpenGL est établi. Un état pour ce plan de recouvrement pour OpenGL est conservé indépendamment de l'état du plan normal de la fenêtre.

4.1 glutEstablishOverlay

La fonction glutEstablishOverlay établit un plan de recouvrement (si possible) pour la *fenêtre courante*.

4.1.1 Usage

```
void glutEstablishOverlay ( void );
```

4.1.2 Description

La fonction glutEstablishOverlay établit un plan de recouvrement (si possible) pour la *fenêtre courante*. Le mode d'affichage requis pour le plan de recouvrement est déterminé par le *mode d'affichage initial*. On peut faire appel à la fonction glutLayerGet(BLUT_OVERLAY_POSSIBLE) pour vérifier si un plan de recouvrement est possible pour la *fenêtre courante* avec le *mode d'affichage initial*. Il ne faut pas tenter d'établir un plan de recouvrement lorsque ce n'est pas possible, GLUT termine le programme.

Si on appelle la fonction glutEstablishOverlay lorsqu'un plan de recouvrement existe déjà, le plan existant est éliminé et un nouveau plan est établi. L'état du plan précédent est éliminé.

4.2 glutUseLayer

La fonction glutUseLayer échange la couche en utilisation pour la *fenêtre courante*.

4.2.1 Usage

```
void glutUseLayer ( GLenum layer );
```

layer Identificateur de la couche soit GLUT_NORMAL ou GLUT_OVERLAY.

4.2.2 Description

La fonction glutUseLayer permet de choisir pour la couche en utilisation entre le plan normal et le plan de recouvrement de la *fenêtre courante*. Le plan de recouvrement doit être choisi seulement s'il existe; cependant, on peut faire appel à glutUseLayer(GLUT_NORMAL) pour une fenêtre sans plan de recouvrement. Les commandes d'OpenGL sont dirigées vers la couche en utilisation.

Pour interroger la couche en utilisation, on peut faire appel à la fonction glutLayerGet(GLUT_LAYER_IN_USE).

4.3 glutRemoveOverlay

La fonction glutRemoveOverlay élimine le plan de recouvrement pour la *fenêtre courante*.

4.3.1 Usage

```
void glutRemoveOverlay ( void );
```

4.3.2 Description

La fonction glutRemoveOverlay élimine le plan de recouvrement pour la *fenêtre courante*. Même si aucun plan de recouvrement existe, il est prudent de faire appel à cette fonction; ceci ne produit aucun effet. Le plan normal est marqué comme la couche en utilisation.

Si l'application prévoit réétablir un plan de recouvrement pour cette fenêtre, il est préférable pour des questions de ressources d'utiliser plutôt les fonctions glutHideOverlay et glutShowOverlay.

4.4 glutPostOverlayRedisplay

Cette fonction indique que le plan de recouvrement de la *fenêtre courante* doit être réaffiché.

4.4.1 Usage

```
void glutPostOverlayRedisplay ( void );
```

4.4.2 Description

Cette fonction indique que le plan de recouvrement de la *fenêtre courante* doit être réaffiché. Lors de la prochaine itération dans la boucle principale de la fonction glutMainLoop, la fonction de rappel pour le plan de recouvrement est appelée pour réafficher le plan de recouvrement de la *fenêtre courante*. Plusieurs appels à cette fonction n'engendrent qu'un seul rafraîchissement du plan de recouvrement. On peut appel à la fonction glutPostOverlayRedisplay à partir d'une fonction de rappel d'affichage ou d'une fonction de rappel d'affichage de plan de recouvrement.

Logiquement, une annonce d'une fenêtre endommagée est traitée comme un appel à glutPostOverlayRedisplay.

4.5 glutShowOverlay et glutHideOverlay

Les fonctions glutShowOverlay et glutHideOverlay montrent ou cachent respectivement le plan de recouvrement pour la *fenêtre courante*.

4.5.1 Usage

```
void glutShowOverlay ( void );

void glutHideOverlay ( void );
```

4.5.2 Description

La fonction glutShowOverlay montre le plan de recouvrement de la *fenêtre courante*. La fonction glutHideOverlay cache le plan de recouvrement de la *fenêtre courante*. L'effet de montrer ou de cacher un plan de recouvrement prend effet immédiatement. Le plan de recouvrement n'est visible qui si le plan normal de la fenêtre est affiché. Il est préférable d'utiliser ces fonctions pour montrer et cacher un plan de recouvrement plutôt que d'éliminer et recréer un plan de recouvrement pour une fenêtre; ceci consomme moins de ressources.

5 Gestion de menus

La bibliothèque GLUT supporte des menus déroulants en cascades. La fonctionnalité est simple et minimale. La bibliothèque GLUT n'a pas la même fonctionnalité que X-Windows ou Windows 95; mais elle a l'avantage d'être portable sur plusieurs plates-formes. Il est illégal de créer ou éliminer des menus, ou de changer, ajouter ou retirer des éléments d'un menu pendant qu'il est en cours d'utilisation.

5.1 glutCreateMenu

La fonction glutCreateMenu crée un nouveau menu déroulant.

5.1.1 Usage

```
int   glutCreateMenu ( void (*func) ( int value ));
```

func La fonction de rappel pour le menu, fonction qui est appelée lorsqu'un élément du menu est sélectionné. La valeur passée à la fonction correspond à la valeur associée à l'élément sélectionné.

5.1.2 Description

La fonction glutCreateMenu crée un nouveau menu déroulant et retourne un entier identifiant ce menu. La plage du numéro de menu commence à 1. Implicitement, le *menu courant* correspond au nouveau menu créé. L'identificateur de menu peut être utilisé par la suite par la fonction glutSetMenu.

Lorsque la fonction de rappel est appelée parce qu'un élément du menu a été sélectionné, la valeur du *menu courant* devient le menu sélectionné. La valeur de la fonction de rappel correspond à l'élément du menu sélectionné.

5.2 glutSetMenu

La fonction glutSetMenu permet d'établir le *menu courant*; la fonction glutGetMenu retourne la valeur du *menu courant.*

5.2.1 Usage

```
void glutSetMenu ( int menu );

int   glutGetMenu ( void );
```

menu L'identificateur du menu pour désigner le *menu courant.*

5.2.2 Description

La fonction glutSetMenu permet d'établir le *menu courant*; la fonction glutGetMenu retourne la valeur du *menu courant.* Si le menu n'existe pas, ou si le *menu courant* précédent a été détruit, glutGetMenu retourne la valeur 0.

5.3 glutDestroyMenu

La fonction glutDestroyMenu détruit un menu spécifique.

5.3.1 Usage

```
void glutDestroyMenu ( int menu );
```

menu L'identificateur du menu pour désigner le *menu courant.*

5.3.2 Description

La fonction glutDestroyMenu détruit le menu identifié par menu. Si menu identifie le *menu courant,* la valeur du *menu courant* devient invalide ou 0.

5.4 glutAddMenuEntry

La fonction glutAddMenuEntry ajoute un élément au bas du *menu courant.*

5.4.1 Usage

```
void glutAddMenuEntry ( char * name, int value );
```

name La chaîne de caractères qui est affichée correspondant à un élément du menu.
value Un identificateur (entier) correspondant à cet élément du menu. Cette valeur est passée à la fonction de rappel lorsqu'un élément du menu est sélectionné.

5.4.2 Description

La fonction glutAddMenuEntry ajoute un élément au bas du *menu courant.* La chaîne de caractères est affichée dans le menu déroulant. Si un élément du menu est sélectionné par un usager, la valeur *value* est la valeur transmise à la fonction de rappel correspondant au *menu courant.*

5.5 glutAddSubMenu

La fonction glutAddSubMenu ajoute une gâchette de sous-menu pour cet élément de menu.

5.5.1 Usage

void glutAddSubMenu (char * name, int menu);

name	La chaîne de caractères qui est affichée correspondant à un élément du menu pour lequel un sous-menu est ouvert en cascade.
menu	Un identificateur (entier) du menu pour lequel un sous-menu est ouvert en cascade.

5.5.2 Description

La fonction glutAddSubMenu ajoute une gâchette de sous-menu pour cet élément de menu. Lors de la sélection de cet élément, un sous-menu menu est ouvert en cascade pour le *menu courant*. Un élément de ce sous-menu peut être par la suite sélectionné.

5.6 glutChangeToMenuEntry

La fonction glutChangeToMenuEntry permet de changer un élément du *menu courant* en une entrée du menu.

5.6.1 Usage

void glutChangeToMenuEntry (int entry, char *name, int value);

entry	Index d'un élément du *menu courant* (1 correspond à l'élément du haut)
name	La chaîne de caractères qui est affichée correspondant à un élément du menu.
value	Un identificateur (entier) correspondant à cet élément du menu. Cette valeur est passée à la fonction de rappel lorsqu'un élément du menu est sélectionné.

5.6.2 Description

La fonction glutChangeToMenuEntry permet de changer un élément du *menu courant* en une entrée du menu. Le paramètre entry indique quel est l'élément du menu qui doit être changé; 1 correspond à l'élément du haut et entry doit être entre 1 et glutGet(GLUT_MENU_NUM_ITEMS) inclusivement. La chaîne de caractères name est affichée pour l'entrée du menu modifiée. Si un élément du menu est sélectionné par un usager, la valeur *value* est la valeur transmise à la fonction de rappel correspondant au *menu courant*.

5.7 glutChangeToSubMenu

La fonction glutChangeToSubMenu permet de changer l'élément du menu du *menu courant* en un élément déclenchant un sous-menu (élément de menu à gâchette).

5.7.1 Usage

void glutChangeToSubMenu (int entry, char *name, int menu);

entry	Index d'un élément du *menu courant* (1 correspond à l'élément du haut)
name	La chaîne de caractères qui est affichée correspondant à un élément du menu pour lequel un sous-menu est ouvert en cascade.
menu	Un identificateur (entier) du menu pour lequel un sous-menu est ouvert en cascade.

5.7.2 Description

La fonction glutChangeToSubMenu permet de changer l'élément du menu du *menu courant* en un élément déclenchant un sous-menu (élément de menu à gâchette). Le paramètre entry indique quel est l'élément du menu qui doit être changé; 1 correspond à l'élément du haut et entry doit être entre

1 et glutGet(GLUT_MENU_NUM_ITEMS) inclusivement. Il n'est pas nécessaire que l'élément du menu à modifier soit un élément à gâchette. L'identificateur menu nomme le menu qui est ouvert en cascade lorsque cet élément est sélectionné.

5.8 glutRemoveMenuItem

La fonction glutRemoveMenuItem élimine un élément du menu.

5.8.1 Usage

void glutRemoveMenuItem (int entry);

entry Index d'un élément du *menu courant* (1 correspond à l'élément du haut) à éliminer.

5.8.2 Description

La fonction glutRemoveMenuItem élimine un élément du menu. Le paramètre entry indique quel est l'élément du menu qui doit être éliminé; 1 correspond à l'élément du haut et entry doit être entre 1 et glutGet(GLUT_MENU_NUM_ITEMS) inclusivement. Les éléments du menu en dessous sont renumérotés.

5.9 glutAttachMenu, glutDetachMenu

Les fonctions glutAttachMenu et glutDetachMenu permettent respectivement d'attacher ou de détacher le *menu courant* à un bouton de la souris.

5.9.1 Usage

void glutAttachMenu (int button);

void glutDetachMenu (int button);

button Le numéro du bouton de la souris.

5.9.2 Description

Ces fonctions attachent ou détachent respectivement le *menu courant* à un des boutons de la souris.

6 Inscription des fonctions de rappel

La bibliothèque GLUT supporte un certain nombre de fonctions de rappel pour répondre aux événements. Il y a trois types de fonctions de rappel:

- **fenêtre :** les fonctions de rappel concernant les fenêtres indiquent quand réafficher ou redimensionner la fenêtre, quand la visibilité de la fenêtre change et quand une entrée est disponible pour la fenêtre;
- **menu :** une fonction de rappel concernant un menu indique la fonction à rappeler lorsqu'un élément du menu est sélectionné;
- **globale :** les fonctions de rappel globales gère le temps et l'utilisation des menus

Les fonctions de rappel attachées à des événements d'entrée doivent être traitées pour les fenêtres pour lesquelles l'événement a été effectué.

6.1 glutDisplayFunc

La fonction glutDisplayFunc établit la fonction de rappel d'affichage pour la *fenêtre courante*.

6.1.1 Usage

```
void glutDisplayFunc ( void (*func) (void));
```

func Identifie la nouvelle fonction de rappel d'affichage.

6.1.2 Description

La fonction glutDisplayFunc établit la fonction de rappel pour la *fenêtre courante*. Quand GLUT détermine que le plan normal de la fenêtre doit être réafficher, la fonction de rappel d'affichage est appelée. Avant l'appel, la *fenêtre courante* devient la fenêtre qui doit être réaffichée et (si aucune fonction de rappel d'affichage du plan de superposition [*overlay*] n'est inscrite) le plan normal devient *la couche en utilisation*. La fonction de rappel d'affichage ne possède aucun paramètre. La région du plan normal entier doit être réaffichée en réponse à la fonction de rappel (incluant les tampons auxiliaires si le programme dépend de leur état).

GLUT détermine quand la fonction de rappel doit être déclenchée en se basant sur l'état d'affichage de la fenêtre. L'état d'affichage peut être modifié explicitement en faisant appel à la fonction glutPostRedisplay ou lorsque le système de fenêtrage rapporte des dommages à la fenêtre. Si plusieurs requêtes d'affichage en différé ont été enregistrées, elles sont regroupées afin de minimiser le nombre d'appel aux fonctions de rappel d'affichage.

Quand un plan de recouvrement (superposition) est établi pour une fenêtre, et qu'aucun fonction de rappel d'affichage du plan de recouvrement de la fenêtre n'est inscrite, la fonction de rappel d'affichage est utilisée pour réafficher les plans normal et de recouvrement de la fenêtre (la fonction est rappelée si l'état du plan normal ou du plan de recouvrement l'exige). Dans ce cas, la *couche en utilisation* n'est pas implicitement changée à l'entrée de la fonction de rappel d'affichage.

Il faut se référer à la documentation de la fonction glutOverlayDisplayFunc pour comprendre la distinction entre les fonctions de rappel des plans normal et de recouvrement d'un fenêtre.

Lorsqu'une fenêtre (ou sous-fenêtre) est créée, aucun fonction de rappel d'affichage n'est inscrite pour cette fenêtre. Chaque fenêtre doit avoir une fonction de rappel inscrite. Une erreur fatale se produit si une tentative d'affichage d'une fenêtre est effectuée sans qu'une fonction de rappel n'ait été inscrite. C'est donc une erreur avec GLUT 3.0 de faire appel à la fonction glutDisplayFunc avec le paramètre NULL.

Au retour de la fonction de rappel, l'état *endommagement normal* de la fenêtre (retourné par la fonction glutLayerGet (GLUT_NORMAL_DAMAGED)) est effacé. Si aucune fonction de rappel pour le plan de superposition de la fenêtre n'est inscrite, l'état *endommagement superposition* (retourné par une appel à la fonction glutLayerGet (GLUT_OVERLAY_DAMAGED))) est également effacé.

6.2 glutOverlayDisplayFunc

La fonction glutOverlayDisplayFunc établit la fonction de rappel d'affichage du plan de recouvrement de la *fenêtre courante*.

6.2.1 Usage

void glutOverlayDisplayFunc (void (*func) (void));

func Identifie la nouvelle fonction de rappel pour l'affichage du plan de recouvrement.

6.2.2 Description

La fonction glutOverlayDisplayFunc établit la fonction de rappel d'affichage du plan de recouvrement de la *fenêtre courante*. La fonction de rappel d'affichage du plan de recouvrement est fonctionnellement la même que la fonction de rappel d'affichage du plan normal, sauf qu'elle affiche le plan de recouvrement.

Quand GLUT détermine que le plan de recouvrement doit être réaffiché, il fait appel à la fonction de rappel d'affichage du plan de recouvrement. Avant l'appel à la fonction de rappel, la *fenêtre courante* devient la fenêtre à réafficher et la *couche en utilisation* devient le plan de recouvrement. La fonction de rappel d'affichage du plan de recouvrement est appelée sans paramètre. La région entière du plan de recouvrement doit être affichée suite à l'appel à la fonction de rappel d'affichage du plan de recouvrement (ceci inclut les tampons auxiliaires si l'application dépend de leur état).

GLUT détermine quand la fonction de rappel du plan de recouvrement doit être déclenchée en se basant sur l'état d'affichage du plan de recouvrement. L'état d'affichage du plan de recouvrement pour la fenêtre courante est explicitement établi par une appel à la fonction glutPostOverlayRedisplay ou implicitement lorsque le système de fenêtrage rapporte que la fenêtre a été endommagée. Plusieurs appels pour un réaffichage du plan de recouvrement sont regroupés pour minimiser la consommation de ressources.

Au retour de la fonction de rappel, l'état *endommagement de recouvrement* de la fenêtre (retourné par la fonction glutLayerGet (GLUT_NORMAL_DAMAGED)) est effacé.

La valeur NULL pour la fonction glutPostOverlayRedisplay désactive la génération de fonction de rappel pour le plan de recouvrement. Quand un plan de recouvrement est créé, la fonction de rappel d'affichage du plan de recouvrement est initialement NULL. Il faut se référer à la fonction glutDisplayFunc pour comprendre que la fonction de rappel d'affichage est utilisée lorsque aucune fonction de rappel d'affichage du plan de recouvrement n'est établie.

6.3 glutReshapeFunc

La fonction glutReshapeFunc établit la fonction de rappel de redimensionnement de la *fenêtre courante*.

6.3.1 Usage

void glutReshapeFunc (void (*func) (int width, int height));

func Identifie la nouvelle fonction de rappel pour le redimensionnement de la fenêtre.

6.3.2 Description

La fonction glutReshapeFunc établit la fonction de rappel de redimensionnement de la *fenêtre courante*. La fonction de rappel de redimensionnement est déclenchée lorsque la fenêtre est refaçonnée. La fonction de rappel est aussi déclenchée immédiatement avant le premier appel à la fonction de rappel d'affichage après la création de la fenêtre ou lorsqu'un plan de recouvrement pour la fenêtre est créé. Les paramètres width et height de la fonction de rappel de redimensionnement spécifient les dimensions en pixels de la nouvelle fenêtre.

Si aucune fonction de rappel de redimensionnement n'est inscrite ou qu'on fait appel à la fonction glutReshapeFunc avec la valeur NULL, la fonction de rappel de redimensionnement implicite est appelée. Cette fonction implicite fait simplement appel à la fonction glViewport(0, 0, width, height) pour le plan normal de la *fenêtre courante*.

Si un plan de recouvrement est établi pour la fenêtre, un appel à la fonction de rappel de redimensionnement est effectué. Il faut se rappeler que c'est la responsabilité de la fonction de redimensionnement de mettre à jour les plans normal et de recouvrement de la *fenêtre courante*.

Lorsqu'un fenêtre racine est refaçonnée, les sous-fenêtres ne sont pas refaçonnées. C'est la responsabilité de 'application de redimensionner les sous-fenêtres et de les positionner à l'intérieur de la fenêtre racine.

6.4 glutKeyboardFunc

La fonction glutKeyboardFunc établit la fonction de rappel pour le clavier pour la *fenêtre courante*.

6.4.1 Usage

void glutKeyboardFunc (void (*func) (unsigned char key, int x, int y);

func Identifie la nouvelle fonction de rappel du clavier pour la fenêtre courante.

6.4.2 Description

La fonction glutKeyboardFunc établit la fonction de rappel du clavier pour la *fenêtre courante*. Lorsqu'un usager tape au clavier (dans une fenêtre), chaque touche génère un appel à la fonction de rappel du clavier. Le paramètre key est le code ASCII de la touche. L'état d'une touche modificatrice telle majuscule [*Shift*] ne peut être connu directement; son effet se reflète cependant sur le caractère ASCII. Les paramètres x et y indiquent les coordonnées relatifs de la souris par rapport à la fenêtre en pixels lors du déclenchement de l'événement (frappe d'une touche). Lors de la création d'une nouvelle fenêtre, aucun fonction de rappel du clavier n'est enregistrée implicitement et les touches du clavier sont ignorées.

La valeur NULL pour la fonction glutKeyboardFunc désactive la génération de fonction de rappel pour le clavier.

Pendant le traitement d'un événement du claviers, on peut faire appel à la fonction glutGetModifiers pour connaître l'état des touches modificatrices (par exemple, la touche majuscule ou Ctrl ou Alt) lors du déclenchement d'un événement au clavier.

Il faut se référer à la fonction glutSpecialFunc pour le traitement de caractères non-ASCII, par exemple les touches de fonction ou les touches fléchées.

6.5 glutMouseFunc

La fonction glutMouseFunc établit la fonction de rappel pour la souris pour la *fenêtre courante*.

6.5.1 Usage

void glutMouseFunc (void (*func) (int button, int state, int x, int y);

func Identifie la nouvelle fonction de rappel lorsqu'un événement est généré à la suite du déclenchement d'une gâchette de la souris.

6.5.2 Description

La fonction glutMouseFunc établit la fonction de rappel de la souris pour la *fenêtre courante*. Lorsqu'un utilisateur appuie ou relâche un des boutons de la souris, chaque gâchette (appui ou relâchement d'un bouton) engendre un appel à la fonction de rappel de la souris.

Le paramètre button peut prendre les valeurs : GLUT_LEFT_BUTTON, GLUT_MIDDLE_BUTTON, ou GLUT_RIGHT_BUTTON. Pour un système à deux boutons, la valeur GLUT_MIDDLE_BUTTON ne peut être générée. Pour un système à un seul bouton, seule la valeur GLUT_LEFT_BUTTON peut être obtenue.

Le paramètre state indique si la fonction de rappel a été rappelée suite à l'appui ou au relâchement d'un bouton de la souris et les valeurs permises sont : GLUT_UP et GLUT_DOWN. Les paramètres x et y indiquent les coordonnées relatifs de la souris par rapport à la fenêtre en pixels lors du déclenchement de l'événement. Lorsqu'un rappel est effectué avec la valeur GLUT_DOWN, l'application peut assumer qu'un rappel sera effectué lorsque le bouton sera relâché (valeur GLUT_UP) (en assumant que la fenêtre possède encore une fonction de rappel enregistrée pour la souris) même si le réticule se trouve en dehors de la fenêtre.

Si un menu est attaché à un bouton de la souris, aucun rappel de la fonction de la souris n'est effectué pour ce bouton.

Pendant le traitement d'un événement de la souris, on peut faire appel à la fonction glutGetModifiers pour connaître l'état des touches modificatrices (par exemple, la touche majuscule ou Ctrl ou Alt) lors du déclenchement d'un gâchette engendrant un rappel de la fonction pour la souris.

La valeur NULL pour la fonction glutMouseFunc désactive la génération de fonction de rappel pour la souris.

6.6 glutMotionFunc, glutPassiveMotionFunc

Les fonctions glutMotionFunc et glutPassiveMotionFunc établissent les fonctions de rappel pour la *fenêtre courante* pour un déplacement de la souris.

6.6.1 Usage

void glutMotionFunc (void (*func) (int x, int y));

void glutPassiveMotionFunc (void (*func) (int x, int y));

func Identifie la nouvelle fonction de rappel pour un déplacement de la souris.

6.6.2 Description

Les fonctions glutMotionFunc et glutPassiveMotionFunc établissent les fonctions de rappel pour la *fenêtre courante* pour un déplacement de la souris. La fonction de rappel spécifiée par glutMotionFunc est appelée lors du déplacement de la souris avec un ou plusieurs boutons appuyés. La fonction de rappel spécifiée par glutPassiveMotionFunc est appelée lors du déplacement de la souris dans la fenêtre avec aucun bouton appuyé. .

Les paramètres x et y indiquent les coordonnées relatifs de la souris par rapport à la fenêtre en pixels.

La valeur NULL pour les fonctions glutMotionFunc ou glutPassiveMotionFunc désactive la génération de fonction de rappel lors du déplacement de la souris.

6.7 glutVisibilityFunc

La fonction glutVisibilityFunc établit la fonction de rappel de visibilité pour la *fenêtre courante*.

6.7.1 Usage

```
void glutVisibilityFunc ( void (*func) (int state ));
```

func Identifie la nouvelle fonction de rappel de visibilité.

6.7.2 Description

La fonction glutVisibilityFunc établit la fonction de rappel de visibilité pour la *fenêtre courante* . Cette fonction de rappel est appelée lorsque la visibilité de la fenêtre change. Le paramètre state peut prendre les valeurs GLUT_VISIBLE ou GLUT_NOT_VISIBLE selon la visibilité de la fenêtre. L'état GLUT_VISIBLE est valable pour une fenêtre partiellement ou totalement visible, i.e. à moins que la visibilité ne change, aucun rafraîchissement de la fenêtre n'est effectué.
GLUT_NOT_VISIBLE signifie donc qu'aucun pixel de la fenêtre ou sous-fenêtre n'est visible.

La valeur NULL pour les fonctions glutVisibilityFunc désactive la fonction de rappel de visibilité.

Si la fonction de rappel de visibilité est désactivée, l'état de la fenêtre devient indéfinie. Tout changement à la visibilité de la fenêtre est rapporté. Donc la réactivation de la fonction de rappel de visibilité vous garantit qu'un changement de visibilité est rapporté.

6.8 glutEntryFunc

La fonction glutEntryFunc établit une fonction de rappel lors de l'entrée ou de la sortie du réticule de la *fenêtre courante.*

6.8.1 Usage

```
void glutEntryFunc ( void (*func) (int state ));
```

func Identifie la nouvelle fonction de rappel lors de l'entrée ou de la sortie du curseur de la souris de la fenêtre courante.

6.8.2 Description

La fonction glutEntryFunc établit une fonction de rappel lors de l'entrée ou de la sortie du réticule de la *fenêtre courante*. Le paramètre state prend les valeurs GLUT_LEFT ou GLUT_ENTERED selon que le curseur de la souris a quitté ou est entré dans la *fenêtre courante*.

La valeur NULL pour les fonctions glutEntryFunc désactive la fonction de rappel d'entré/sortie de la fenêtre. Certains systèmes de fenêtrage peuvent générer des valeurs erronées.

6.9 glutSpecialFunc

La fonction glutSpecialFunc établit la fonction de rappel pour les caractères non-ASCII provenant du clavier.

6.9.1 Usage

void glutSpecialFunc (void (*func) (int key, int x, int y));

func Identifie la nouvelle fonction de rappel pour les caractères non-ASCII du clavier.

6.9.2 Description

La fonction glutSpecialFunc établit la fonction de rappel du clavier pour les caractères non-ASCII pour la *fenêtre courante*. Des caractères non-ASCII sont générés du clavier lorsqu'une des touches de fonction (F1 à F12) ou une des touches de direction est utilisée.

Le paramètre key est une constante correspondant à une touche spéciale (GLUT_KEY_*). Les paramètres x et y indiquent les coordonnées relatifs de la souris par rapport à la fenêtre en pixels lors du déclenchement d'une gâchette au clavier.

Pendant le traitement d'un événement du clavier, on peut faire appel à la fonction glutGetModifiers pour connaître l'état des touches modificatrices (par exemple, la touche majuscule ou Ctrl ou Alt) lors du déclenchement d'un gâchette engendrant un rappel de la fonction pour le clavier (touches spéciales).

La valeur NULL pour la fonction glutSpecialFunc désactive la génération de fonction de rappel pour le clavier (touches spéciales).

Les valeurs correspondant aux touches spéciales sont les suivantes:

GLUT_KEY_F1	Touche F1.
GLUT_KEY_F2	Touche F2.
GLUT_KEY_F3	Touche F3.
GLUT_KEY_F4	Touche F4.
GLUT_KEY_F5	Touche F5.
GLUT_KEY_F6	Touche F6.
GLUT_KEY_F7	Touche F7.
GLUT_KEY_F8	Touche F8.
GLUT_KEY_F9	Touche F9.
GLUT_KEY_F10	Touche F10.
GLUT_KEY_F11	Touche F11.
GLUT_KEY_F12	Touche F12.
GLUT_KEY_LEFT	Touche fléchée vers la gauche.
GLUT_KEY_UP	Touche fléchée vers le haut.
GLUT_KEY_RIGHT	Touche fléchée vers la droite.
GLUT_KEY_DOWN	Touche fléchée vers le bas.
GLUT_KEY_PAGE_UP	Touche page précédente (**Page up**).

GLUT_KEY_PAGE_DOWN	Touche page suivante (**Page down**).
GLUT_KEY_HOME	Touche **Home.**
GLUT_KEY_END	Touche **End.**
GLUT_KEY_INSERT	Touche d'insertion (**ins**);

Il est à noter que les touches d'échappement [*Escape*], de recul [*Backspace*] et d'élimination [*delete*] génèrent des caractères ASCII. Voici quelques valeurs importantes de caractères ASCII:

Touche	Valeur en décimal
Recul [*Backspace*]	8
Tabulation [*Tab*]	9
A la ligne [*Linefeed -- CTRL_Enter*]	10
Entrée [*Enter*]	13
Échappement [*Escape*]	27
Élim.[*Delete*]	127

6.10 glutSpaceballMotionFunc

6.10.1 Usage

void glutSpaceballMotionFunc (void (*func) (int x, int y, int z));

func Identifie la nouvelle fonction de rappel pour l'affichage..

6.11 glutSpaceballRotateFunc

6.11.1 Usage

void glutSpaceballRotateFunc (void (*func) (int x, int y, int z));

func Identifie la nouvelle fonction de rappel pour l'affichage.

6.12 glutSpaceballButtonFunc

6.12.1 Usage

void glutSpaceballButtonFunc (void (*func) (int button, int state));

func Identifie la nouvelle fonction de rappel pour l'affichage.

6.13 glutButtonBoxFunc

6.13.1 Usage

void glutButtonBoxFunc (void (*func) (int button, int state));

func Identifie la nouvelle fonction de rappel pour l'affichage.

6.14 glutDialsFunc

La fonction glutDialsFunc établit la fonction de rappel des valuateurs pour la *fenêtre courante*.

6.14.1 Usage

void glutDialsFunc (void (*func) (int dial, int value));

func Identifie la nouvelle fonction de rappel des valuateurs.

6.14.2 Description

La fonction glutDialsFunc établit la fonction de rappel des valuateurs pour la *fenêtre courante*. Le paramètre dial est le numéro du rhéostat (commençant à 1). On peut obtenir le nombre de rhéostats disponibles en faisant appel à la fonction glutDeviceGet(GLUT_NUM_DIALS). Le paramètre value donne la valeur de la rotation absolue en degrés. La valeur value n'est pas réinitialisée après une rotation complète; mais la valeur continue de s'accroître de façon continue jusqu'à un débordement de value.

Dans le cas où un poste ne serait pas muni de valuateurs, l'inscription d'une fonction de rappel n'a aucune influence et ne provoque aucune erreur. La fonction de rappel n'est simplement pas appelée.

La valeur NULL pour la fonction glutDialsFunc désactive la génération de fonction de rappel pour les valuateurs. Initialement, lorsqu'une nouvelle fenêtre est créée, aucune fonction de rappel pour les valuateurs n'est inscrite.

6.15 glutTabletMotionFunc

6.15.1 Usage

void glutTabletMotionFunc (void (*func) (int x, int y));

func Identifie la nouvelle fonction de rappel pour l'affichage..

6.16 glutTabletButtonFunc

6.16.1 Usage

void glutTabletButtonFunc (void (*func) (int button, int state, int x, int y));

func Identifie la nouvelle fonction de rappel pour l'affichage..

6.17 glutMenuStatusFunc, glutMenuStateFunc

La fonction glutMenuStatusFunc établit une fonction de rappel pour l'état du menu.

6.17.1 Usage

void glutMenuStatusFunc (void (*func) (int status, int x, int y));

void glutMenuStateFunc (void (*func) (int status));

func Identifie la nouvelle fonction de rappel pour l'état du menu.

6.17.2 Description

La fonction glutMenuStatusFunc établit une fonction de rappel pour l'état du menu de sorte qu'une application utilisant GLUT puisse déterminer si le menu est en utilisation ou non. Quand une fonction de rappel d'état du menu est inscrite, un appel est effectué avec la valeur GLUT_MENU_IN_USE pour le paramètre status quand les menus déroulants sont utilisés; la valeur GLUT_MENU_NOT_IN_USE pour le paramètre status est utilisée lorsque les menus ne sont pas en utilisation. Les paramètres x et y indique la position, en coordonnées de fenêtre, du réticule lorsque le menu a été déclenché par un bouton de la souris. Le paramètre func représente la fonction de rappel. Les autres fonctions de rappel (excepté les fonctions de rappel pour le déplacement de la souris) continuent à être actives pendant l'utilisation des menus, de sorte que la fonction de rappel pour l'état du menu peut suspendre une animation ou d'autres tâches lorsque le menu est en cours d'utilisation. Une cascade de sous-menus pour une menu initial déroulant ne génère pas d'appel à la fonction de rappel pour l'état du menu. Il y a une seule fonction de rappel pour l'état du menu dans GLUT.

Quand la fonction de rappel de l'état du menu est appelée, le *menu courant* devient le menu initial déroulant dans les cas GLUT_MENU_IN_et GLUT_MENU_NOT_IN_USE. La *fenêtre courante* devient la fenêtre de laquelle le menu initial a surgi, également dans les deux cas.

La valeur NULL pour la fonction glutMenuStatusFunc désactive la génération de fonction de rappel pour l'état du menu.

La fonction glutMenuStateFunc est une version obsolète de la fonction glutMenuStatusFunc.

6.18 glutIdleFunc

La fonction glutIdleFunc établit la fonction de rappel au repos (il n'y a pas d'autres activités).

6.18.1 Usage

```
void glutIdleFunc ( void (*func) ( void ));
```

func Identifie la nouvelle fonction de rappel pour le repos.

6.18.2 Description

La fonction glutIdleFunc établit la fonction de rappel au repos de telle sorte que GLUT peut effectuer des tâches de traitement à l'arrière plan ou effectuer une animation continue lorsque aucun événement n'est reçu. La fonction de rappel n'a aucun paramètre. Cette fonction est continuellement appelé lorsque aucun événement n'est reçu. La *fenêtre courante* et le *menu courant* ne sont pas changés avant l'appel à la fonction de rappel. Les applications utilisant plusieurs fenêtres ou menus doivent explicitement établir *fenêtre courante* et le *menu courant*, et ne pas se fier à l'état courant.

On doit éviter les calculs dans une fonction de rappel pour le repos afin de minimiser les effets sur le temps de réponse interactif.

6.19 glutTimerFunc

La fonction glutTimerFunc établit une fonction de rappel de minuterie qui est appelée dans un nombre déterminé de millisecondes.

6.19.1 Usage

void glutTimerFunc (unsigned int msecs, void (*func) (int value), value);

msecs	Le nombre de millisecondes avant de faire appel à la fonction de rappel.
func	Identifie la nouvelle fonction de rappel de minuterie.
value	La valeur associée à la minuterie.

6.19.2 Description

La fonction glutTimerFunc établit une fonction de rappel de minuterie qui est appelée dans un nombre déterminé de millisecondes. La valeur du paramètre value de la fonction de rappel est la valeur du paramètre de la fonction glutTimerFunc. Plusieurs appels de la fonction de rappel à la même heure ou à des heures différentes peuvent être inscrits simultanément.

Le nombre de millisecondes constitue une borne inférieure avant qu'un appel à la fonction de rappel soit effectué. GLUT essaie d'effectuer l'appel à la fonction de rappel aussitôt que possible après l'expiration du délai.

Il n'y a aucun moyen pour annuler une inscription d'une fonction de rappel de minuterie. Il faut plutôt ignorer l'appel en se basant sur la valeur du paramètre value.

7 Gestion des index de couleur

OpenGL supporte les modes RGBA et le mode d'index de couleurs. Le mode RGBA est généralement préféré, parce qu'il offre plus de flexibilité et que le mode indexé nécessite de charger une table d'index de couleurs.

Les fonctions d'index de couleur de GLUT sont utilisées pour écrire et lire des entrées dans la table de couleurs de la fenêtre. Chaque fenêtre de GLUT en mode indexé possède sa propre table d'index de couleurs. La taille de la table est déterminée par la fonction glutGet(GLUT_COLORMAP_SIZE).

Les fenêtres en mode indexé dans une application peuvent partager les ressources d'index en copiant un index de couleur dans plusieurs fenêtres par la fonction glutCopyColormap. Si possible, GLUT tente de partager la table d'index actuelle. Copier un index de couleur avec glutCopyColormap peut permettre de partager physiquement les table d'index de couleurs, même si logiquement, chaque fenêtre possède sa propre table d'index. Pour cette raison, les applications doivent généralement charger une table d'index de couleurs dans une fenêtre, et par la suite copier tous les index de couleurs dans les autres fenêtres et ne modifier aucune cellule (index) de la table.

L'utilisation de plusieurs index de couleurs peut engendrer certains problèmes d'affichage.

7.1 glutSetColor

La fonction glutSetColor spécifie la couleur d'une entrée de la table d'index pour la *couche en utilisation* de la *fenêtre courante*.

7.1.1 Usage

void glutSetColor (int cell, GLfloat red, GLfloat green, GLfloat blue);

cell	Identifie l'index de couleurs de la cellule.
red	Intensité du rouge (entre 0.0 et 1.0).
green	Intensité du vert (entre 0.0 et 1.0).
Blue	Intensité du bleu (entre 0.0 et 1.0).

7.1.2 Description

La fonction glutSetColor spécifie la couleur d'une entrée de la table d'index pour la *couche en utilisation* de la *fenêtre courante*. Les valeurs valides de l'index sont : $0 <$ cell $<$ nombre d'index. Les valeurs des composantes doivent entre comprises entre 0.0 et 1.0.

7.2 glutGetColor

La fonction glutGetColor retourne la valeur d'une composante de couleur pour un index donné pour la *couche en utilisation* de la *fenêtre courante.*

7.2.1 Usage

GLfloat glutGetColor (int cell, int component);

cell	Identifie l'index de couleurs de la cellule.
Component	Une des composantes GLUT_RED, GLUT_GREEN ou GLUT_BLUE.

7.2.2 Description

La fonction glutGetColor retourne la valeur d'une composante de couleur rouge, vert ou bleu pour un index donné pour la *couche en utilisation* de la *fenêtre courante*. La *fenêtre courante* doit être en mode indexé. Le paramètre cell doit être dans l'intervalle $0 \leq$ cell $<$ nombre d'index. Pour des valeurs d'index correctes, les valeurs retournées sont entre 0.0 et 1.0. glutSetColor retourne la valeur -1 si la valeur de l'index est un index de recouvrement transparent ou l'index < 0 ou $\geq$ glutGet(GLUT_WINDOW_COLORMAP_SIZE).

7.3 glutCopyColormap

La fonction glutCopyColormap copie logiquement la table de couleurs d'une fenêtre donnée dans la *couche en utilisation* de la *fenêtre courante*.

7.3.1 Usage

void glutCopyColormap (int win);

win	Identifie la fenêtre de laquelle la table des index de couleurs doit être copiée.

7.3.2 Description

La fonction glutCopyColormap copie logiquement la table de couleurs d'une fenêtre donnée dans la *couche en utilisation* de la *fenêtre courante*. La copie est effectuée du plan normal au plan normal ou du plan de recouvrement au plan de recouvrement. Une fois la table de couleurs copiée, il faut éviter de modifier des cellules avec la fonction glutSetColor, ce qui force une copie effective (actuelle) de la table si elle avait été précédemment copiée par référence. La fonction

glutCopyColormap doit être appelée seulement si la fenêtre win et la fenêtre courante sont en mode indexé.

8 Obtenir l'état

La bibliothèque GLUT contient un grand nombre de variables d'état dont un certain nombre (pas toutes) peut être interrogé directement.

8.1 glutGet

La fonction glutGet retourne des entiers représentant l'état de GLUT.

8.1.1 Usage

int glutGet (GLenum state);

state Identificateur de la variable d'état.

Les états de GLUT sont:

État	**Description**
GLUT_WINDOW_X	Position en *x* en pixels relative à l'origine de la *fenêtre courante*.
GLUT_WINDOW_Y	Position en *y* en pixels relative à l'origine de la *fenêtre courante*.
GLUT_WINDOW_WIDTH	Largeur en pixels de la *fenêtre courante*.
GLUT_WINDOW_HEIGHT	Hauteur en pixels de la *fenêtre courante*.
GLUT_WINDOW_BUFFER_SIZE	Nombre total de bits du tampon de couleur de la *fenêtre courante*. Pour une fenêtre RGBA, c'est la somme de GLUT_WINDOW_RED_SIZE, GLUT_WINDOW_GREEN_SIZE, GLUT_WINDOW_BLUE_SIZE, et GLUT_WINDOW_ALPHA_SIZE. Pour une fenêtre en mode indexé, il s'agit de la taille des index de couleur.
GLUT_WINDOW_STENCIL_SIZE	Nombre total de bits du tampon du pochoir de la *fenêtre courante*.
GLUT_WINDOW_DEPTH_SIZE	Nombre total de bits du tampon de profondeur de la *fenêtre courante*.
GLUT_WINDOW_RED_SIZE	Nombre total de bits de la couleur rouge dans le tampon de couleur de la *fenêtre courante*. Zéro si la fenêtre utilise un index de couleur.
GLUT_WINDOW_GREEN_SIZE	Nombre total de bits de la couleur vert dans le tampon de couleur de la *fenêtre courante*. Zéro si la fenêtre utilise un index de couleur.
GLUT_WINDOW_BLUE_SIZE	Nombre total de bits de la couleur bleu dans le tampon de couleur de la *fenêtre courante*. Zéro si la fenêtre utilise un index de couleur.
GLUT_WINDOW_ALPHA_SIZE	Nombre total de bits de la composante α (alpha) dans le tampon de couleur de la *fenêtre courante*. Zéro si la fenêtre utilise un index de couleur.
GLUT_WINDOW_ACCUM_RED_SIZE	Nombre total de bits de la couleur rouge dans le tampon d'accumulation de la *fenêtre courante*. Zéro si la fenêtre utilise un index de couleur.
GLUT_WINDOW_ACCUM_GREEN_SIZE	Nombre total de bits de la couleur vert dans le tampon d'accumulation de la *fenêtre courante*. Zéro si

	la fenêtre utilise un index de couleur.
GLUT_WINDOW_ACCUM_BLUE_SIZE	Nombre total de bits de la couleur bleu dans le tampon d'accumulation de la *fenêtre courante*. Zéro si la fenêtre utilise un index de couleur.
GLUT_WINDOW_ACCUM_ALPHA_SIZE	Nombre total de bits de la composante α (alpha) dans le tampon d'accumulation de la *fenêtre courante*. Zéro si la fenêtre utilise un index de couleur.
GLUT_WINDOW_DOUBLEBUFFER	1 si la fenêtre a un double tampon; 0 sinon.
GLUT_WINDOW_RGBA	1 si la fenêtre est en mode RGBA, 0 sinon.
GLUT_WINDOW_PARENT	Le numéro de la fenêtre parent; 0 s'il s'agit d'un fenêtre racine.
GLUT_WINDOW_NUM_CHILDREN	Le nombre de sous-fenêtres que la *fenêtre courante* possède (en ne comptant les enfants des enfants).
GLUT_WINDOW_COLORMAP_SIZE	Taille de la table des index de couleurs de la *fenêtre courante*; 0 si le mode RGBA est utilisé.
GLUT_WINDOW_NUM_SAMPLES	Nombre d'échantillons de la *fenêtre courante* si le mode multiéchantillonnage est utilisé.
GLUT_WINDOW_STEREO	1 si la fenêtre courante est stéréo, 0 dans le cas contraire.
GLUT_WINDOW_CURSOR	Le curseur courante de la *fenêtre courante*.
GLUT_SCREEN_WIDTH	Indique la largeur de l'écran en pixels; 0 indique que la largeur est inconnue ou non disponible.
GLUT_SCREEN_HEIGHT	Indique la hauteur de l'écran en pixels; 0 indique que la hauteur est inconnue ou non disponible.
GLUT_SCREEN_WIDTH_MM	Largeur initiale en millimètres de la *fenêtre courante*. 0 indique que cette valeur est inconnue ou non disponible.
GLUT_SCREEN_HEIGHT_MM	Hauteur initiale en millimètres de la *fenêtre courante*. 0 indique que cette valeur est inconnue ou non disponible.
GLUT_MENU_NUM_ITEMS	Le nombre d'élément dans le *menu courant*.
GLUT_DISPLAY_MODE_POSSIBLE	Signifie si le *mode courant d'affichage* est supporté.
GLUT_INIT_DISPLAY_MODE	Le masque du *mode initial d'affichage*.
GLUT_INIT_WINDOW_X	Position initiale en *x* en pixels relative à l'origine de la *fenêtre courante*.
GLUT_INIT_WINDOW_Y	Position initiale en *y* en pixels relative à l'origine de la *fenêtre courante*.
GLUT_INIT_WINDOW_WIDTH	Largeur initiale en pixels de la *fenêtre courante*.
GLUT_INIT_WINDOW_HEIGHT	Hauteur initiale en pixels de la *fenêtre courante*.
GLUT_ELPASED_TIME	Nombre de millisecondes depuis l'appel à glutlinit ou depuis le premier appel à glutGet(GLUT_ELAPSED_TIME).

8.1.2 Description

La fonction glutGet interroge les variables d'état représenté par des entiers de la bibliothèque GLUT. Le paramètre state détermine quel état doit être retourné. Ces variables d'état de la fenêtre correspondent à la *couche en utilisation* de la fenêtre. Les variables d'état dont le nom commence par GLUT_WINDOW_ retournent des valeurs correspondant à la *fenêtre courante*. Les variables d'état dont le nom commence par GLUT_MENU retourne des valeurs concernant le *menu courant*. Les autres variables correspondent à des états globaux. Si une requête est incorrecte, la valeur -1 est retournée.

8.2 glutLayerGet

La fonction glutLayerGet retourne l'état d'une des couches de la *fenêtre courante*.

8.2.1 Usage

int glutLayerGet (GLenum info);

Info Identifie le type d'information désirée. Les valeurs permises sont:

GLUT_OVERLAY_POSSIBLE	Vrai si un plan de recouvrement peut être établi pour la *fenêtre courante* avec le mode courant *d'affichage initial*. Si c'est faux, un appel à la fonction glutEstablishOverlay engendre une erreur fatale.
GLUT_LAYER_IN_USE	Soit GLUT_NORMAL ou GLUT_OVERLAY si la *couche en utilisation* est soit le plan normal ou le plan de recouvrement de la *fenêtre courante*.
GLUT_HAS_OVERLAY	1 si la *fenêtre courante* possède un plan de recouvrement; 0 dans le cas contraire.
GLUT_TRANSPARENT_INDEX	Index de couleur de transparence pour le plan de recouvrement de la *fenêtre courante*, -1 s'il n'y a pas de plan de recouvrement.
GLUT_NORMAL_DAMAGED	Vrai si le plan normal de la *fenêtre courante* a été endommagé (par l'activité du système) depuis le dernier appel à la fonction de rappel d'affichage. S'il y a eu un appel à glutPostRedisplay et l'affichage est en attente, la valeur retournée est faux.
GLUT_OVERLAY_DAMAGED	Vrai si le plan de recouvrement de la *fenêtre courante* a été endommagé (par l'activité du système) depuis le dernier appel à la fonction de rappel d'affichage du plan de recouvrement. S'il y a eu un appel à glutPostRedisplay ou à la fonction glutPostOverlayRedisplay et l'affichage est en attente, la valeur retournée est faux. -1 s'il n'y a pas de plan de recouvrement.

8.2.2 Description

La fonction glutLayerGet permet d'obtenir différentes couches d'information de GLUT pour la *fenêtre courante*. Le paramètre info indique quel type d'information doit être retourné.

8.3 glutDeviceGet

La fonction glutDeviceGet permet d'obtenir certaines information concernant les dispositifs disponibles sur le poste de travail.

8.3.1 Usage

int glutDeviceGet (GLenum info);

Info Identifie l'information d'un dispositif à obtenir. Les valeurs permises sont:

GLUT_HAS_KEYBOARD	Différent de zéro si le poste possède un clavier, zéro dans le cas contraire.
GLUT_HAS_MOUSE	Différent de zéro si le poste possède une souris, zéro dans le cas contraire.
GLUT_HAS_SPACEBALL	
GLUT_HAS_DIAL_AND_BUTTON_BOX	

GLUT_HAS_TABLET	Différent de zéro si le poste possède une tablette graphique, zéro dans le cas contraire.
GLUT_NUM_MOUSE_BUTTONS	Le nombre de boutons de la souris, 0 s'il n'y a pas de souris.
GLUT_NUM_SPACEBALL_BUTTONS	Le nombre de boutons supportés par la souris 3D *[spaceball]*.
GLUT_NUM_BUTTON_BOX_BUTTONS	Le nombre de boutons supportés par le dispositif valuateur. . La valeur 0 est retournée s'il n'y a pas de dispositif valuateur.
GLUT_NUM_DIALS	Le nombre de rhéostats supportés par le dispositif valuateur. La valeur 0 est retournée s'il n'y a pas de dispositif valuateur.
GLUT_NUM_TABLET_BUTTONS	Le nombre de boutons de la tablette graphique, 0 s'il n'y a pas de tablette graphique.

8.3.2 Description

La fonction glutDeviceGet permet d'obtenir certaines information concernant les dispositifs disponibles sur le poste de travail. Cette information est représentée par des entiers. Une valeur -1 est retournée par la fonction si un appel à la fonction demande des informations incorrectes.

8.4 glutGetModifiers

La fonction glutGetModifiers permet d'obtenir les touches modificatrices du clavier ou des boutons de la souris.

8.4.1 Usage

```
int glutGetModifiers ( void );
```

Les valeurs retournées par cette fonction sont:

GLUT_ACTIVE_SHIFT	Une des touches modificatrices **Shift** ou **CapsLock.**
GLUT_ACTIVE_CTRL	La touche modificatrice **Ctrl.**
GLUT_ACTIVE_ALT	La touche modificatrice **Alt.**

8.4.2 Description

La fonction glutGetModifiers retourne la valeur d'une des touches modificatrices lorsqu'un événement d'entrée est généré à partir du clavier, d'une touche spéciale ou de la souris. On ne doit faire appel à cette fonction que lors du traitement d'une fonction de rappel du clavier, des touches spéciales ou de la souris. Le système de fenêtrage peut intercepter certaines touches modificatrices; dans ce cas, aucun appel à des fonctions de rappel n'est effectué.

8.5 glutExtensionsSupported

8.5.1 Usage

```
int glutExtensionsSupported ( char *extension );
```

9 Rendu des polices de caractères

La bibliothèque GLUT supporte deux types de polices de caractères: les polices haute qualité [*stroke fonts*] pour lesquelles chaque caractère est construit à l'aide de segments de lignes et les polices de basse qualité [*bitmap fonts*] qui sont formées d'un ensemble de pixels et affichées avec la fonction glbitmap. Les polices haute qualité ont l'avantage de pouvoir être mises à l'échelle. Les polices basse qualité sont moins flexibles mais habituellement plus rapide à afficher.

9.1 glutBitmapCharacter

9.1.1 Usage

void glutBitmapCharacter (void *font, int character);

font	Identifie la police de caractères.
character	Le caractère à afficher.

9.1.2 Description

Sans aucune liste d'affichage, la fonction glutBitmapCharacter affiche le caractère *character* selon la police de caractères *font.* Les polices de caractères disponibles sont:

GLUT_BITMAP_8_BY_13	Une police de caractères de largeur fixe où chaque caractère est contenu dans un rectangle de 8 par 13 pixels. Sous le système X, le nom de cette police est: -misc-fixed-medium-r-normal--13-120-75-75-C-80-iso8859-1
GLUT_BITMAP_9_BY_15	Une police de caractères de largeur fixe où chaque caractère est contenu dans un rectangle de 9 par 15 pixels. Sous le système X, le nom de cette police est: -misc-fixed-medium-r-normal--15-140-75-75-C-90-iso8859-1
GLUT_BITMAP_TIMES_ROMAN_10	Une police de caractères de 10 points à espacement proportionnel romaine. Sous le système X, le nom de cette police est: -adobe-times-medium-r-normal--10-100-75-75-p-54-iso8859-1
GLUT_BITMAP_TIMES_ROMAN_24	Une police de caractères de 24 points à espacement proportionnel romaine. Sous le système X, le nom de cette police est: -adobe-times-medium-r-normal--24-240-75-75-p-124-iso8859-1
GLUT_BITMAP_HELVETICA_10	Une police de caractères de 10 points à espacement proportionnel helvétique. Sous le système X, le nom de cette police est: -adobe-helvetica-medium-r-normal--10-100-75-75-p-56-iso8859-1
GLUT_BITMAP_HELVETICA_12	Une police de caractères de 12 points à espacement proportionnel helvétique. Sous le système X, le nom de cette police est: -adobe-helvetica-medium-r-normal--12-120-75-75-p-67-iso8859-1
GLUT_BITMAP_HELVETICA_18	Une police de caractères de 18 points à espacement proportionnel helvétique. Sous le système X, le nom de cette police est: -adobe-helvetica-medium-r-normal--18-180-75-75-p-98-iso8859-1

Pour une chaîne de caractères, on utilise la fonction glutBitmapCharacter dans une boucle pour la longueur de la chaîne. Pour se positionner pour le premier caractère de la chaîne, on utilise la fonction glRasterPos2f.

9.2 glutBitmapWidth

Cette fonction retourne la largeur en pixel d'un caractère.

9.2.1 Usage

int glutBitmapWidth (GLUTbitmapFont font, int character);

font	Identifie la police de caractères de basse qualité.
character	Le caractère pour lequel on retourne la largeur en pixels.

9.2.2 Description

La fonction glutBitmapWidth retourne en pixels, la largeur d'un caractère dans une police de caractères supportée. Pendant que la largeur d'une police de caractères peut varier (la largeur d'une police fixe ne varie pas), la taille maximum d'une police est toujours fixe.

9.3 glutStrokeCharacter

Cette fonction affiche un caractère de haute qualité.

9.3.1 Usage

void glutStrokeCharacter (void * font, int character);

font	Identifie la police de caractères de haute qualité.
character	Le caractère à afficher.

9.3.2 Description

En n'utilisant aucune liste d'affichage, le caractère *character* est affiché selon la police de caractères *font*. Les polices de caractères disponibles sont:

GLUT_STROKE_ROMAN	Une police de caractères à espacement proportionnel romaine simplexe pour les caractères ASCII de 32 à 127.
GLUT_STROKE_MONO_ROMAN	Une police de caractères à espacement fixe romaine simplexe pour les caractères ASCII de 32 à 127.

La fonction glTranslatef est utilisée pour positionner le premier caractère d'une chaîne de texte.

9.4 glutStrokeWidth

Cette fonction retourne la largeur en pixels d'un caractère.

9.4.1 Usage

void glutStrokeWidth (GLUTstrokeFont font, int character);

font	Identifie la police de caractères de basse qualité.

character Le caractère pour lequel on retourne la largeur en pixels.

9.4.2 Description

La fonction glutStrokeWidth retourne en pixels, la largeur d'un caractère dans une police de caractères supportée. Pendant que la largeur d'une police de caractères peut varier (la largeur d'une police fixe ne varie pas), la taille maximum d'une police est toujours fixe.

10 Rendu d'objets géométriques

10.1 glutSolidSphere, glutWireSphere

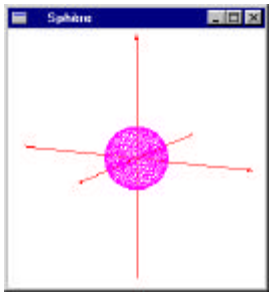
Les fonctions glutSolidSphere et glutWireSphere affichent une sphère pleine ou en fil de fer.

10.1.1 Usage

```
void glutSolidSphere ( GLdouble radius, GLint slices, GLint stacks );

void glutWireSphere ( GLdouble radius, GLint slices, GLint stacks );
```

radius Le rayon de la sphère.
slices Le nombre de subdivisions autour de l'axe de *z* (similaire à la longitude).
stacks Le nombre de subdivisions le long de l'axe de *z* (similaire à la latitude).

10.1.2 Description



Affichent une sphère centrée à l'origine de rayon *radius*. La sphère est subdivisée en tranches et en pile autour et le long de l'axe des *z*.

10.2 glutSolidCube, glutWireCube

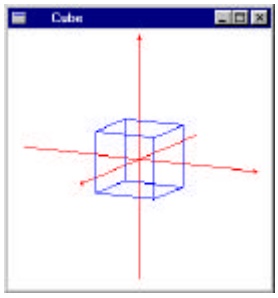
Les fonctions glutSolidCube et glutWireCube affichent un cube plein ou en fil de fer.

10.2.1 Usage

```
void glutSolidCube ( GLdouble size );

void glutWireCube ( GLdouble size );
```

Size Le largeur du côté du cube en coordonnées de modélisation.

10.2.2 Description



Affichent un cube plein ou en fil de fer centré à l'origine. La largeur du côté est donnée par *Size*.

10.3 glutSolidCone, glutWireCone

Les fonctions glutSolidCone et glutWireCone affichent respectivement un cône plein ou en fil de fer.

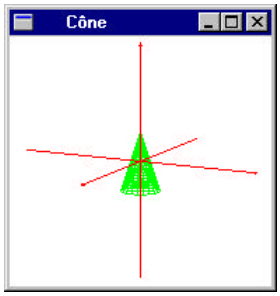
10.3.1 Usage

void glutSolidCone (GLdouble base, GLdouble height, GLint slices, GLint stacks);

void glutWireCone (GLdouble base, GLdouble height, GLint slices, GLint stacks);

base	Le rayon de la base du cône.
height	La hauteur du cône.
slices	Le nombre de subdivisions autour de l'axe de z (similaire à la longitude).
stacks	Le nombre de subdivisions le long de l'axe de *z* (similaire à la latitude).

10.3.2 Description



Affichent un cône plein ou en fil de fer. La base est à *z=0* et le sommet du cône est à *z=height*. Le cône est subdivisé en tranches autour de l'axe des *z* et en pile le long de l'axe des *z*.

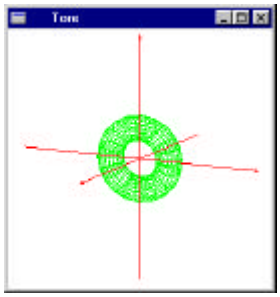
10.4 glutSolidTorus, glutWireTorus

10.4.1 Usage

void glutSolidTorus (GLdouble innerRadius, GLdouble outerRadius, GLint nsides, GLint rings);

void glutWireTorus (GLdouble innerRadius, GLdouble outerRadius, GLint nsides, GLint rings);

10.4.2 Description



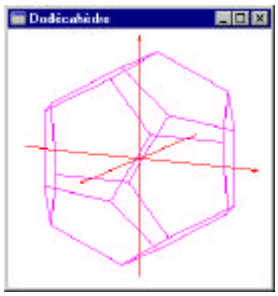
Affichent un tore plein ou en fil de fer. Le rayon intérieur innerRadius est utilisé pour calculer une section de cercle qui tourne autour du rayon extérieur outerRadius. Le tore est composé de rings anneaux subdivisées en nsides côtés.

10.5 glutSolidDodecahedron, glutWireDodecahedron

10.5.1 Usage

void glutSolidDodecahedron (void);

void glutWireDodecahedron (void);

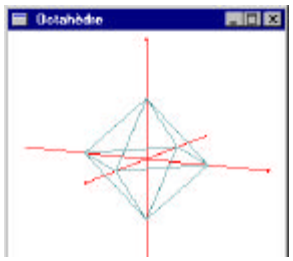


10.5.2 Description

Affichent un dodécaèdre (12 côtés réguliers) plein ou en fil de fer centré à l'origine de rayon $\sqrt{3}$ en coordonnées de modélisation.

10.6 glutSolidOctahedron, glutWireOctahedron

10.6.1 Usage



```
void glutSolidOctahedron ( void );

void glutWireOctahedron ( void );
```

10.6.2 Description

Affichent un octaèdre plein ou en fil de fer centré à l'origine de rayon 1 en coordonnées de modélisation.

10.7 glutSolidTetrahedron, glutWireTetrahedron

10.7.1 Usage

```
void glutSolidTetrahedron ( void );

void glutWireTetrahedron ( void );
```

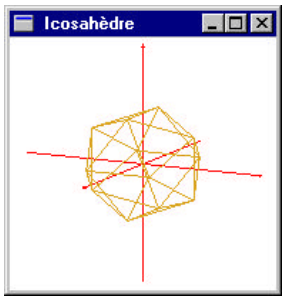
10.7.2 Description

Affichent un tétraèdre plein ou en file de fer centré à l'origine de rayon $\sqrt{3}$ en coordonnées de modélisation.

10.8 glutSolidIcosahedron, glutWireIcosahedron

10.8.1 Usage

```
void glutSolidIcosahedron ( void );

void glutWireIcosahedron ( void );
```



10.8.2 Description

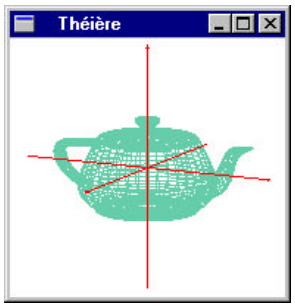
Affichent un icosaèdre plein ou en file de fer centré à l'origine de rayon 1.0 en coordonnées de modélisation.

10.9 glutSolidTeapot, glutWireTeapot

Les fonctions glutSolidTeapot et glutWireTeapot affichent une théière[1] pleine ou en fil de fer.

10.9.1 Usage

```
void glutSolidTeapot ( void );

void glutWireTeapot ( void );
```



10.9.2 Description

[1] Il s'agit de la fameuse théière modélisée par Martin Newell en 1975.

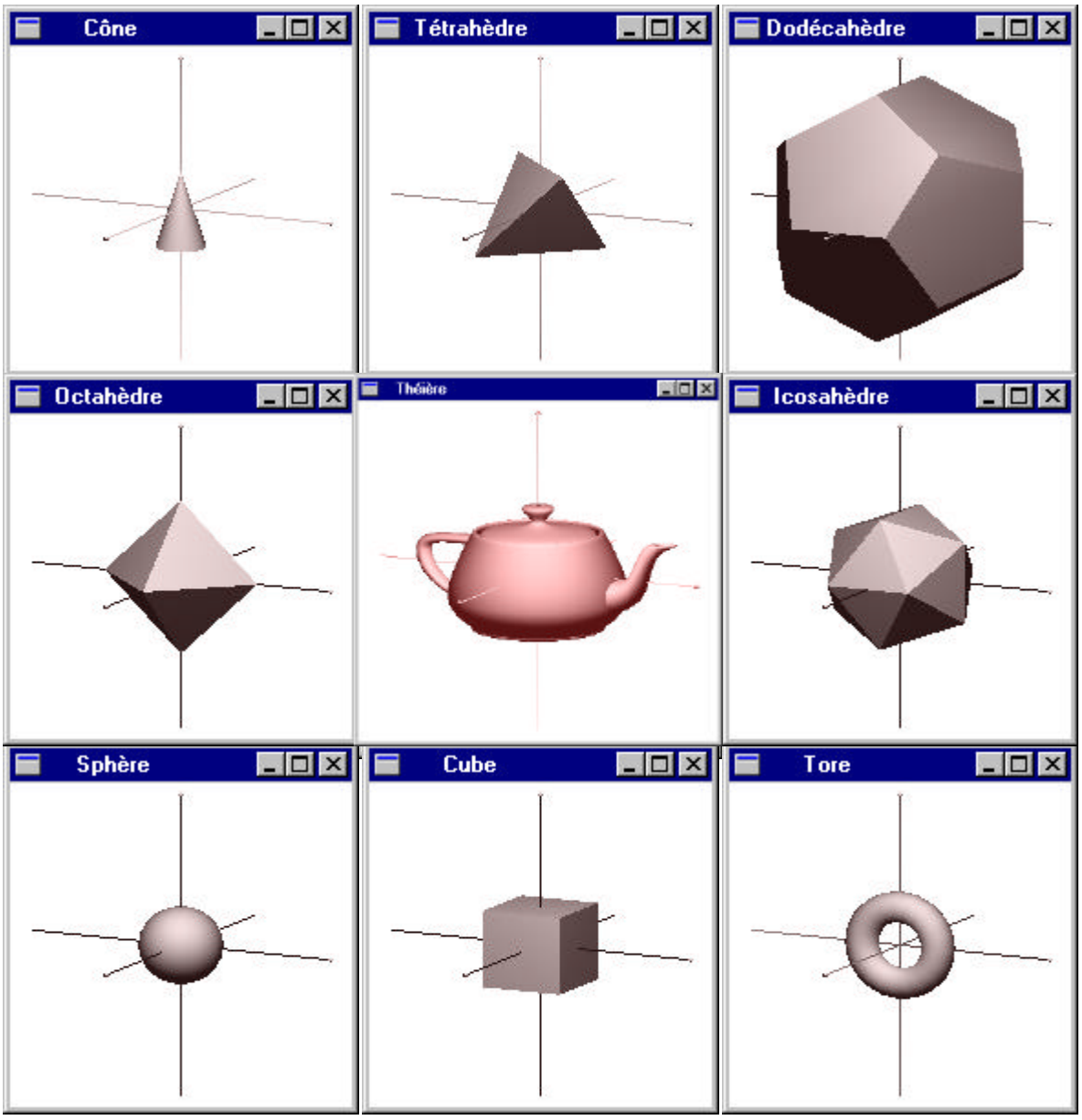
Ces fonctions affichent respectivement une théière pleine ou en fil de fer. Les coordonnées pour les normales à la surface et les coordonnées pour les textures sont générées. La théière est générée par les évaluateurs d'OpenGL.

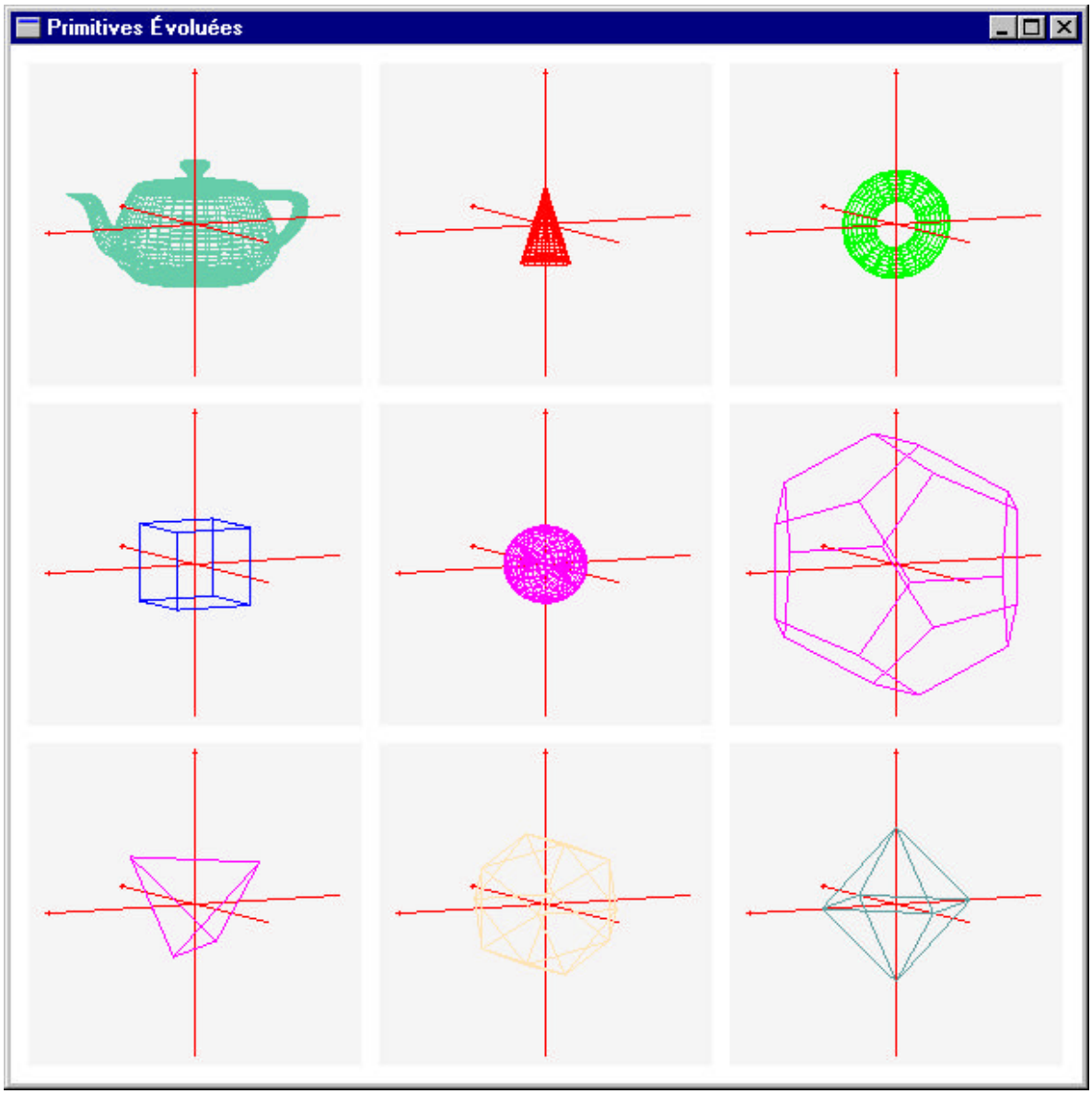
```
with glut, GL, GLU;
with Ada.Characters.Latin_1;
with Ada.Text_IO ;
with Ada.Numerics.Elementary_Functions ;
with Ada.Exceptions ;
with rgbcolor;
with System;
with tg;
procedure FenêtreMultiples is
      use gl, glut ;
      use rgbcolor;
      use Ada.Text_IO;
      GAP        : constant Integer := 10 ;
      main_w, w1, w2, w3, w4 : Integer ;
      Fin_Anormale : exception;

      procedure Reshape ( w, h : in GLsizei);
      pragma Convention ( StdCall, Reshape );

      procedure AfficherFenêtre;
      procedure AfficherThéière;
      procedure AfficherCône;
      procedure AfficherTore;
      procedure AfficherCube;
      procedure AfficherSphère;
      procedure AfficherDodécahèdre;
      procedure AfficherTétrahèdre;
      procedure AfficherIcosahèdre;
      procedure AfficherOctahèdre;


      pragma Convention ( StdCall, AfficherFenêtre);
      pragma Convention ( StdCall, AfficherThéière);
      pragma Convention ( StdCall, AfficherCône);
      pragma Convention ( StdCall, AfficherTore);
      pragma Convention ( StdCall, AfficherCube);
      pragma Convention ( StdCall, AfficherSphère);
      pragma Convention ( StdCall, AfficherDodécahèdre);
      pragma Convention ( StdCall, AfficherTétrahèdre);
      pragma Convention ( StdCall, AfficherIcosahèdre);
      pragma Convention ( StdCall,AfficherOctahèdre);

      width : Integer:= 90;
      height : Integer:= 90;
      DL_Axe3D : gl.uint;

      type Fenêtre_Type is
      record
            Numéro          : Natural := 1;
                  PosInit         : GL.Coord_int.Poi
                  Largeur         : Integer := 100;
                  Hauteur         : Integer := 100;
                  Couleur         : GL.Color_Float.C
                  FoncAfficher : System.Address;
                  Titre : String ( 1 ..15);
            end record;
            Fenêtres : array ( Integer range <>) of
 ( 1 => ( 1, ( 10, 10), 90, 90, RGB_Value.White,
Théière     "),
                  2 => ( 2, (110, 10),      90, 90, I
            AfficherCône'Address,           "    Côr
                  3 => ( 3, (210, 10),      90, 90, I
            AfficherTore'Address,           "    Tore
                  4 => ( 1, ( 10, 110),     90, 90, I
            AfficherCube'Address,           "    Cub
             5 => ( 2, (110, 110),          90, 90, I
            AfficherSphère'Address,         "    Sphe
                  6 => ( 3, (210, 110),     90, 90, I
            AfficherDodécahèdre'Address,"Dodécal
                  7 => ( 1, ( 10, 210),     90, 90, I
           AfficherTétrahèdre'Address,      " Tétrah
                  8 => ( 2, (110, 210),     90, 90, I
            AfficherIcosahèdre'Address,     " Icosah
                  9 => ( 3, (210, 210),     90, 90, I
            AfficherOctahèdre'Address,      " Octahe

              );


            function "+" ( P1, P2 : gl.Coord_Float.F
            begin
                  return (P1.x + P2.x, P1.y + P2.y,
            end "+";


            function "-" ( P1, P2 : gl.Coord_Float.F
            begin
                  return (P1.x - P2.x, P1.y - P2.y, F
            end "-";
            procedure TracerAxe3D ( Origine : gl.C
                                    longueur :GL
                                    Couleur : GL.
            begin
                  glCallList ( DL_Axe3D);
            end TracerAxe3D;
```

```
      procedure CréerAxe3D ( DL_Axe3D : out GL.uint;
                             Origine     : in gl.Coord_Float.Point3 := (0.0, 0.0, 0.0);
                             longueur    : in GL.Float;
                             Couleur     : in GL.Color_Float.Color3 :=
RGB_Value.Red)  is
            Px : gl.Coord_Float.Point3 := Origine + (longueur, 0.0, 0.0);
            Py : gl.Coord_Float.Point3 := Origine + (0.0, longueur, 0.0);
            Pz : gl.Coord_Float.Point3 := Origine + (0.0, 0.0, longueur);
      begin
            DL_Axe3D := gl.GenLists ( 1 );
            gl.NewList ( DL_Axe3D, GL.COMPILE );
                  gl.PushMatrix;
                  tg.Line ( Origine - (Px.x, Px.y, Px.z), Px, Couleur);
                  tg.Line ( Origine - (Py.x, Py.y, Py.z), Py, Couleur);

                  tg.Line ( Origine - (Pz.x, Pz.y, Pz.z), Pz, Couleur);
                  gl.Translatef ( 0.0, 0.0, longueur);
                  glut.SolidCone ( 0.03, 0.075, 10, 10 );

                  gl.Translatef ( 0.0, longueur, -longueur);
                  gl.Rotatef    ( -90.0, 1.0, 0.0, 0.0);
                  glut.SolidCone ( 0.03, 0.075, 10, 10 );

                  gl.Translatef ( longueur, 0.0, -1.0 * longueur);
                  gl.Rotatef ( +90.0, 1.0, 0.0, 0.0);
                  gl.Rotatef ( -270.0, 0.0, 1.0, 0.0);

                  glut.SolidCone ( 0.03, 0.075, 10, 10 );
                  gl.PopMatrix;
            gl.EndList;
      end  CréerAxe3D;

      procedure Initialiser is
      begin
      --    gl.Enable ( GL.DEPTH_TEST );
            CréerAxe3D ( DL_Axe3D, (0.0, 0.0, 0.0), 1.95 );
      end Initialiser;

      procedure Reshape ( w, h : in GLsizei) is
            Index : Integer := 0;
            FenêtreCourante : Integer := glut.GetWindow;
      begin
       glViewport(0, 0, w, h);
       if (w > 50) then
          width := (Integer(w) - 4 * GAP) / 3;
       else
          width := 10;
       end if ;
       if (h > 50) then
          height := (Integer(h) - 4 * GAP) / 3;
       else
          height := 10;
       end if ;

       for i in Fenêtres'Range loop
            glut.SetWindow
            glut.PositionWindow      (((i-1) m
                              width, ((i-1) / 3) *h
            glut.ReshapeWindow     (width, h
            gl.ClearColor                ((Fenêtr
            gl.LoadIdentity;
            glu.LookAt     ((1.0, 0.5, 2.0), (0.

            gl.MatrixMode                (GL_PR
            gl.LoadIdentity;

            if (w <= h) then
               gl.Ortho(-2.0, 2.0, -2.0 *Gldou
                              2.0 * Gldouble (h)
            else
               gl.Ortho(-2.0, 2.0, -2.0 *Gldou
                              2.0 *Gldouble (w
            end if ;
            gl.MatrixMode(GL_MODELVIEW
       end loop;
      end Reshape;

      procedure AfficherFenêtre is
      begin
            glClear(GL_COLOR_BUFFER_B
            glFlush;

      end AfficherFenêtre;

      procedure  AfficherThéière is
      begin
            glClear(GL_COLOR_BUFFER_B
               glColor ( RGB_Value.Mediu
               glut.WireTeapot ( 1.0 );
               TracerAxe3D ((0.0, 0.0, 0.0),
            glFlush;
      end AfficherThéière;

      procedure  AfficherCône is
      begin
            glClear(GL_COLOR_BUFFER_B
```

```
            glColor ( RGB_Value.Green );
            gl.PushMatrix;
            gl.Translatef ( 0.0, -0.5, 0.0 );
            gl.Rotatef ( -90.0, 1.0, 0.0, 0.0);
            glut.WireCone ( 0.3, 1.0, 20, 20 );
            gl.PopMatrix;
            TracerAxe3D ((0.0, 0.0, 0.0), 1.5);
        glFlush;
end AfficherCône;

procedure AfficherTore is
begin
        glClear(GL_COLOR_BUFFER_BIT);
            glColor ( RGB_Value.Green );
            glut.WireTorus ( 0.2, 0.5, 20, 20 );
            TracerAxe3D ((0.0, 0.0, 0.0), 1.5);
        glFlush;
end AfficherTore;

procedure AfficherCube is
begin
        glClear(GL_COLOR_BUFFER_BIT);
            TracerAxe3D ((0.0, 0.0, 0.0), 1.5);
            glColor ( RGB_Value.Blue );
            glut.WireCube ( 1.0 );
        glFlush;
end AfficherCube;

procedure AfficherSphère is
begin
        glClear(GL_COLOR_BUFFER_BIT);
            TracerAxe3D ((0.0, 0.0, 0.0), 1.5);
            glColor ( RGB_Value.Magenta );
            glut.WireSphere ( 0.5, 20, 20 );
        glFlush;
end AfficherSphère;

procedure AfficherDodécahèdre is
begin
        glClear(GL_COLOR_BUFFER_BIT);
            TracerAxe3D ((0.0, 0.0, 0.0), 1.9);
            glColor ( RGB_Value.Magenta );
            glut.WireDodecahedron;
        glFlush;
end AfficherDodécahèdre;

procedure AfficherTétrahèdre is
begin
        glClear(GL_COLOR_BUFFER_BIT);
                TracerAxe3D ((0.0, 0.0, 0.0),
                glColor ( RGB_Value.Magent
                glut.WireTetrahedron;
            gl.Flush;
        end AfficherTétrahèdre;

        procedure AfficherIcosahèdre is
        begin
            glClear(GL_COLOR_BUFFER_E
                TracerAxe3D ((0.0, 0.0, 0.0),
                gl.Color ( RGB_Value.Golder
                glut.WireIcosahedron;
            glFlush;
        end AfficherIcosahèdre;

        procedure AfficherOctahèdre is
        begin
            gl.Clear(GL_COLOR_BUFFER_
                TracerAxe3D ((0.0, 0.0, 0.0),
                gl.Color ( RGB_Value.CadetE
                glut.WireOctahedron;
            gl.Flush;
        end AfficherOctahèdre;

begin
  glut.Init;
  glut.InitDisplayMode(GLUT.GLUT_RGB);
  glut.InitWindowPosition ( 50, 50 );
  glut.InitWindowSize ( 610, 610);

  main_w := glut.CreateWindow ("Fenêtres M
  glut.DisplayFunc                (Affiche
  glut.ReshapeFunc                (Reshap
  gl.ClearColor                   (RGB_V

  for i in Fenêtres'Range loop
      glut.InitDisplayMode(GLUT.GLUT_RGE
      glut.InitWindowPosition (Integer(Fenêtr
      Integer(Fenêtres(i).PosInit.y));
      glut.InitWindowSize ( Fenêtres(i).largeu

      Fenêtres(i).Numéro := glut.CreateWind
      glut.DisplayFunc            (Fenêtre
      gl.ClearColor               (Fenêtre
      Initialiser;
  end loop;
  glut.MainLoop ;
exception
   when Constraint_Error =>
```

```
        Put_Line ("Constraint_error ---------") ;

    when Nasty_Occurrence : others =>
        Put_Line ("Exception dans FenêtreMultiples " &
Ada.Exceptions.Exception_Name (Nasty_Occurrence)) ;
        Put_Line ("Exception dans FenêtreMultiples " &
Ada.Exceptions.Exception_Information (Nasty_Occurrence)) ;
        Put_Line ("Exception dans FenêtreMultiples " &
Ada.Exceptions.Exception_Message ( Nasty_Occurrence)) ;

end FenêtreMultiples ;
```


Cône
Tétrahèdre
Dodécahèdre
Octahèdre
Théière
Icosahèdre
Sphère
Cube
Tore


Primitives Évoluées