NAME

glutIdleFunc - sets the global idle callback.

SYNTAX

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

ARGUMENTS

func	The new idle callback function.

DESCRIPTION

glutIdleFunc sets the global idle callback to be func so a GLUT program can perform background processing tasks or continuous animation when window system events are not being received. If enabled, the idle callback is continuously called when events are not being received. The callback routine has no parameters. The current window and current menu will not be changed before the idle callback. Programs with multiple windows and/or menus should explicitly set the current window and/or current menu and not rely on its current setting.

The amount of computation and rendering done in an idle callback should be minimized to avoid affecting the program's interactive response. In general, not more than a single frame of rendering should be done in an idle callback.

Passing NULL to glutIdleFunc disables the generation of the idle callback.

EXAMPLE

A typical idle callback to animate a window might look like:

```
void
idle(void)
{
  time += 0.05;
  glutSetWindow(window);
  glutPostRedisplay();
}
```

Notice how the idle callback does not do any actual drawing; it only advances the time scene state global variable. That is left to the window's display callback which will be triggered by the call to glutPostRedisplay.

If you use the idle callback for animation, you should be sure to stop rendering when the window is not visible. This is easy to set up with a visibility callback. For example:

```
void
visible(int vis)
{
  if (vis == GLUT_VISIBLE)
    glutIdleFunc(idle);
  else
    glutIdleFunc(NULL);
}
```

If you do use the idle callback for animation, one thing you should *not* do is setup the idle callback before calling glutMainLoop. It is much better to use the visibility callback to install idle callback when the window first becomes visible on the screen.

SEE ALSO

glutTimerFunc, glutVisibilityFunc

AUTHOR

Mark J. Kilgard (mjk@sgi.com)