GLUT provides a way for you to register the function that will be responsible for processing events generated by mouse clicks. The name of this function is *glutMouseFunc*,and it is commonly called in the initialization phase of the application. The syntax is as follows:

Usage

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

`func`

The new mouse callback function.

As we can see from the signature of *glutMouseFunc*, the function that will handle the mouse click events must have four parameters.

The first relates to which button was pressed, or released. This argument can have one of three values:

- GLUT_LEFT_BUTTON
- GLUT_MIDDLE_BUTTON
- GLUT_RIGHT_BUTTON

The second argument relates to the state of the button when the callback was generated, i.e. pressed or released. The possible values are:

- GLUT_DOWN
- GLUT_UP

When a callback is generated with the state GLUT_DOWN, the application can assume that a GLUT_UP will come afterwards even if the mouse moves outside the window.

The remaining two parameters provide the (*x*,*y*) coordinates of the mouse relatively to the upper left corner of the client area of the window.

Detecting Motion

GLUT provides mouse motion detection capabilities to an application. There are two types of motion that GLUT handles: active and passive motion. Active motion occurs when the mouse is moved and a button is pressed. Passive motion is when the mouse is moving but no buttons are pressed. If an application is tracking motion, an event will be generated per frame during the period that the mouse is moving.

As usual you must register with GLUT the function that will be responsible for handling the motion events. GLUT allows us to specify two different functions: one for tracking passive motion, and another to track active motion.

The signatures for the GLUT functions are as follows:

---

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

Parameters:

- func – the function that will be responsible for the respective type of motion.

The parameters for the motion processing function are the (*x*,*y*) coordinates of the mouse relatively to the upper left corner of the window's client area.

Detecting when the mouse enters or leaves the window

GLUT is also able to detect when the mouse leaves or enters the window area. A callback function can be registered to handle these two events. The GLUT function to register this callback is *glutEntryFunc* and the syntax is as follows:

---

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

Parameters:

- func – the function that will handle these events.

The parameter of the function that will handle these events tells us if the mouse has entered of left the window region. GLUT defines two constants that can be used in the application:

- GLUT_LEFT
- GLUT_ENTERED

Note: This doesn't work exactly as it says in Microsoft Windows, this is because in Microsoft's OS the focus is changed with a mouse click. Although you can change this is your own system using some tools from Microsoft, others are likely to have the standard setting so its probably better if you don't use this feature in Microsoft Windows to detect when the mouse enters/leaves the window.

Here are some common use cases for the GLUT mouse function:

1. **Navigating the Scene**:
    - **Rotation**: Rotate the 3D scene or object based on mouse movements, often by holding down a mouse button and dragging.
    - **Panning**: Move the scene horizontally or vertically.
    - **Zooming**: Zoom in or out of the scene, often using the scroll wheel or by holding a button and moving the mouse.
2. **Object Selection and Interaction**:

- **Picking**: Select objects in the scene by clicking on them.
- **Dragging and Dropping**: Move objects within the scene by clicking and dragging them to a new location.
- **Highlighting**: Change the appearance of an object when the mouse hovers over it or clicks on it.

3. **User Interface Controls**:
   - **Buttons**: Create interactive buttons in the window that respond to mouse clicks.
   - **Sliders**: Adjust values using draggable sliders.
   - **Menus**: Implement context-sensitive menus that appear upon right-clicking.
4. **Drawing Applications**:
   - **Sketching**: Draw lines, shapes, or other primitives directly on the screen by clicking and dragging.
   - **Editing**: Modify existing drawings by selecting and manipulating them with the mouse.
5. **Games**:
   - **Camera Control**: Control the game camera's orientation or position.
   - **Player Interaction**: Handle player actions such as shooting, selecting units, or moving characters.
   - **Inventory Management**: Drag and drop items in a game inventory system.
6. **3D Modeling Tools**:
   - **Vertex Manipulation**: Select and move vertices, edges, or faces of 3D models.
   - **Tool Selection**: Switch between different tools or modes (e.g., move, scale, rotate) using mouse clicks.

References :

- https://www.opengl.org/resources/libraries/glut/spec3/node50.html
- http://www.lighthouse3d.com/tutorials/glut-tutorial/the-mouse/