

```

1  using System;
2  using System.Linq;
3  using System.Collections.Generic;
4  using System.Threading;
5  using System.Threading.Tasks;
6  using System.IO;
7  using Structures;
8  using static Program.Constants;
9  using Gtk;
10 using Gdk;
11 using Cairo;
12 using Graphics;
13 using static Program.Program;
14 namespace Program {
15     static class Input {
16         private static bool canMove = false;
17         private static Vector3 rootPos = null;
18         private static Vector3 rootAngle = null;
19         private static readonly double MOUSE_SENSITIVITY = 1;
20         private static readonly double SCROLL_SENSITIVITY = 1.1;
21         private static readonly double TIME_SENSITIVITY = 1.2;
22         private static readonly double RADIUS_SENSITIVITY = 1.1;
23         private static readonly int LINE_SENSITIVITY = 5;
24         private static double focal_length = -1;
25         [GLib.ConnectBefore]
26         public static void OnKeyPress(object sender, KeyPressEventArgs args) {
27             if (args.Event.Key == Gdk.Key.f) {
28                 if (Program.activesys == null) return;
29                 else {
30                     Program.activesys.IterateCenter();
31                     Program.sys_view.ClearPaths();
32                 }
33                 args.RetVal = true;
34             } else if (args.Event.Key == Gdk.Key.r) {
35                 double d = Vector3.Magnitude
36 (Program.sys_view.camera.position);
37                 Program.sys_view.camera = new Camera(d, Vector3.zero);
38             } else if (args.Event.Key == Gdk.Key.l) {
39                 canMove = !canMove;
40                 if (!canMove) {
41                     rootPos = null;
42                 }
43             } else if (args.Event.Key == Gdk.Key.Up) {
44                 Program.sys_view.radius_multiplier *= RADIUS_SENSITIVITY;
45             } else if (args.Event.Key == Gdk.Key.Down) {
46                 Program.sys_view.radius_multiplier /= RADIUS_SENSITIVITY;
47             } else if (args.Event.Key == Gdk.Key.Right) {
48                 Program.activesys.Stop();
49                 Program.timestep *= TIME_SENSITIVITY;
50                 Program.activesys.StartAsync(step: Program.timestep);
51             } else if (args.Event.Key == Gdk.Key.Left) {
52                 Program.activesys.Stop();
53                 Program.timestep /= TIME_SENSITIVITY;
54                 Program.activesys.StartAsync(step: Program.timestep);
55             } else if (args.Event.Key == Gdk.Key.Page_Down) {
56                 // don't make it smaller than 0
57                 if (Program.sys_view.line_max >= LINE_SENSITIVITY) {
58                     Program.sys_view.line_max -= LINE_SENSITIVITY;
59                 }
60             } else if (args.Event.Key == Gdk.Key.Page_Up) {
61                 Program.sys_view.line_max += LINE_SENSITIVITY;
62             } else if (args.Event.Key == Gdk.Key.Escape) {
63                 Program.sys_view.Stop();
64                 Program.activesys.Stop();
65                 Program.mainWindow.Destroy();

```

```

66         var menu = new UI.Menu();
67         var data = new UI.SaveData() {
68             bodies = ((IEnumerable<Body>)Program.activesys).ToList(),
69             centers = Program.CustomCenters,
70             timestep = Program.timestep,
71             radius_multiplier = Program.sys_view.radius_multiplier,
72             line_max = Program.sys_view.line_max
73         };
74         menu.temp_savedata = data;
75         menu.loadButton.Click();
76     } else if (args.Event.Key == Gdk.Key.q) {
77         Program.sys_view.camera = new Camera(Vector3.Magnitude
78 (Program.sys_view.camera.position)*SCROLL_SENSITIVITY,Program.sys_view.camera.angle);
79     } else if (args.Event.Key == Gdk.Key.w) {
80         Program.sys_view.camera = new Camera(Vector3.Magnitude
81 (Program.sys_view.camera.position)/
82 SCROLL_SENSITIVITY,Program.sys_view.camera.angle);
83     } else if (args.Event.Key == Gdk.Key.c) {
84         if (focal_length == -1) {
85             Console.WriteLine("hi");
86             focal_length = Vector3.Magnitude
87 (Program.sys_view.camera.position);
88         Program.sys_view.camera = new Camera
89 (1000*AU,Program.sys_view.camera.angle);
90         //Program.sys_view.ClearPaths();
91         //Program.sys_view.Redraw();
92     } else {
93         Console.WriteLine("hi2");
94         Program.sys_view.camera = new Camera
95 (focal_length,Program.sys_view.camera.angle);
96         //Program.sys_view.Redraw();
97         focal_length = -1;
98     }
99 }
100 }
101 [GLib.ConnectBefore]
102 public static void OnMouseMovement(Object sender,
103 MotionNotifyEventArgs args) {
104     if (canMove) {
105         if (rootPos == null || rootAngle == null ) {
106             rootPos = new Vector3(args.Event.X,args.Event.Y,0);
107             rootAngle = Program.sys_view.camera.angle;
108         } else {
109             double d = Vector3.Magnitude
110 (Program.sys_view.camera.position);
111             Program.sys_view.camera = new Camera(d,rootAngle +
112 deg*MOUSE_SENSITIVITY* new Vector3(rootPos.y - args.Event.Y,0,args.Event.X -
113 rootPos.x));
114         } args.RetVal = true;
115     }
116 }
117 [GLib.ConnectBefore]
118 public static void OnScrollMovement(Object sender, ScrollEventArgs
119 args) {
120     if (args.Event.Direction == Gdk.ScrollDirection.Up) {
121         Program.sys_view.bounds_multiplier /= SCROLL_SENSITIVITY;
122         Program.sys_view.camera = new Camera(Vector3.Magnitude
123 (Program.sys_view.camera.position)/
124 SCROLL_SENSITIVITY,Program.sys_view.camera.angle);
125     } else if (args.Event.Direction == Gdk.ScrollDirection.Down) {
126         Program.sys_view.bounds_multiplier *= SCROLL_SENSITIVITY;
127         Program.sys_view.camera = new Camera(Vector3.Magnitude
128 (Program.sys_view.camera.position)*SCROLL_SENSITIVITY,Program.sys_view.camera.angle);
129     }
130 }

```

```
118     }  
119 }
```