```
using System;
using System.Linq;
using System.Collections.Generic;
using System.Threading;
using System.Threading.Tasks;
using System.IO;
using Structures;
using static Program.Constants;
using Gtk;
using Gdk;
using Cairo;
using Graphics;
using static Program.Program;
namespace Program {
    static class Input {
        private static bool canMove = false;
        private static Vector3 rootPos = null;
        private static Vector3 rootAngle = null;
        public static readonly double mouse_sensitivity = 1;
        public static readonly double scroll_sensitivity = 1.1;
        public static readonly double time_sensitivity = 1.2;
        public static readonly double radius_sensitivity = 1.1;
        public static readonly int line_sensitivity = 5;
        [GLib.ConnectBefore]
        public static void KeyPress(object sender, KeyPressEventArgs args) {
                if (args.Event.Key == Gdk.Key.f) {
                        if (Program.activesys == null) return;
                            else {
                        Program.activesys.IterateCenter();
                        Program.sys_view.ClearPaths();
                    }
                    args.RetVal = true;
                } else if (args.Event.Key == Gdk.Key.r) {
                    double d = Vector3.Magnitude(Program.sys_view.camera.position);
                    Program.sys_view.camera = new Camera(d,Vector3.zero);
                } else if (args.Event.Key == Gdk.Key.l) {
                    canMove = !canMove;
                    if (!canMove) {
                        rootPos = null;
                    }
                } else if (args.Event.Key == Gdk.Key.Up) {
                    Program.sys_view.radius_multiplier *= radius_sensitivity;
                } else if (args.Event.Key == Gdk.Key.Down) {
                    Program.sys_view.radius_multiplier /= radius_sensitivity;
                } else if (args.Event.Key == Gdk.Key.Right) {
                    Program.activesys.Stop();
                    Program.timestep *= time_sensitivity;
                    Program.activesys.StartAsync(step: Program.timestep);
                } else if (args.Event.Key == Gdk.Key.Left) {
                    Program.activesys.Stop();
                    Program.timestep /= time_sensitivity;
                    Program.activesys.StartAsync(step: Program.timestep);
                } else if (args.Event.Key == Gdk.Key.Page_Down) {
                    // don't make it smaller than 0
                    if (Program.sys_view.line_max >= line_sensitivity) {
                        Program.sys_view.line_max -= line_sensitivity;
                    }
                } else if (args.Event.Key == Gdk.Key.Page_Up) {
                    Program.sys_view.line_max += line_sensitivity;
                } else if (args.Event.Key == Gdk.Key.Escape) {
                    Program.sys_view.Stop();
                    Program.activesys.Stop();
                    Program.mainWindow.Destroy();

                    var menu = new UI.Menu();
                    var data = new UI.SaveData() {
```

```
67                      bodies = ((IEnumerable<Body>)Program.activesys).ToList(),
68                      centers = Program.CustomCenters,
69                      timestep = Program.timestep,
70                      radius_multiplier = Program.sys_view.radius_multiplier,
71                      line_max = Program.sys_view.line_max
72                  };
73                  menu.temp_savedata = data;
74                  menu.loadButton.Click();
75              }
76
77              }
78          [GLib.ConnectBefore]
79          public static void MouseMovement(Object sender, MotionNotifyEventArgs
    args) {
80              if (canMove) {
81                  if (rootPos == null || rootAngle == null ) {
82                      rootPos = new Vector3(args.Event.X,args.Event.Y,0);
83                      rootAngle = Program.sys_view.camera.angle;
84                  } else {
85                      double d = Vector3.Magnitude
    (Program.sys_view.camera.position);
86                      Program.sys_view.camera = new Camera(d,rootAngle +
    deg*mouse_sensitivity* new Vector3(rootPos.y - args.Event.Y,args.Event.X -
    rootPos.x, 0));
87                  } args.RetVal = true;
88              }
89          }
90          [GLib.ConnectBefore]
91          public static void Scroll(Object sender, ScrollEventArgs args) {
92              if (args.Event.Direction == Gdk.ScrollDirection.Up) {
93                  Program.sys_view.bounds_multiplier /= scroll_sensitivity;
94              } else if (args.Event.Direction == Gdk.ScrollDirection.Down) {
95                  Program.sys_view.bounds_multiplier *= scroll_sensitivity;
96              }
97          }
98      }
99  }
```