```
using System;
using System.IO;
using System.Linq;
using System.Collections.Generic;
using Gtk;
using Cairo;
using static Program.Program;
using static Program.Constants;
using Structures;
namespace UI {
    public class Menu : Window {
        protected VBox containerbox;
        protected VBox controlbox;
        protected ScrolledWindow systemscrollbox;
        protected VBox systembox;
        protected HBox donebox;
        protected Scale TimestepScale;
        protected Scale RScale;
        protected Scale LineScale;
        protected ComboBoxText BodyCombo;
        public Button loadButton {get; set;}
        protected Entry filename;
        protected readonly String SYSTEM_DIRECTORY = "ExampleSystems";
        internal List<BodyBox> new_bodies {get; set;} = new List<BodyBox>();
        public SaveData temp_savedata {get; set;} = null;
        protected static List<bool> centers = new List<bool>();

        public Menu(Gtk.WindowType s = Gtk.WindowType.Toplevel) : base(s) {
            this.SetDefaultSize(300,400);
            this.DeleteEvent += delegate { Application.Quit (); };
            containerbox = new VBox(homogeneous: false, spacing: 3);
            controlbox = new VBox(homogeneous: false, spacing: 3);
            systemscrollbox = new ScrolledWindow();
            systembox = new VBox(homogeneous: false, spacing: 3);
            donebox = new HBox(homogeneous: false, spacing: 3);

            var l1 = new Label("Mechanics Timestep");
            var l2 = new Label("Planetary Radii Multiplier");
            var l3 = new Label("Orbit Trail Length");
            TimestepScale = new Scale(Orientation.Horizontal, 0.1,1000,0.1);
            TimestepScale.Value = 50;
            RScale = new Scale(Orientation.Horizontal, 1, 1000, 1);
            RScale.Value = 100;
            LineScale = new Scale(Orientation.Horizontal, 50, 1000, 1);
            LineScale.Value = 100;

            var addBox = new HBox();
            var addButton = new Button("Add");
            addButton.Clicked += new EventHandler(OnAddClick);
            var filenameText = new Label("Save File: ");
            filename = new Entry();
            var saveButton = new Button("Save");
            saveButton.Clicked += new EventHandler(OnSaveClick);
            loadButton = new Button("Load");
            loadButton.Clicked += new EventHandler(OnLoadClick);
            var helpButton = new Button("?");
            helpButton.Clicked += new EventHandler(OnHelpClick);
            BodyCombo = new ComboBoxText();
            BodyCombo.AppendText("Custom");
            foreach (Body b in Examples.solar_system_bodies) {
                BodyCombo.AppendText(b.name);
            }
            BodyCombo.Active = 0; // Default to Custom body
            addBox.PackStart(BodyCombo, true, false, 3);
            addBox.PackStart(addButton, true, false, 3);
            addBox.PackStart(filenameText, true, false, 3);
```

```
67              addBox.PackStart(filename, true, false, 3);
68              addBox.PackStart(saveButton, true, false, 3);
69              addBox.PackStart(loadButton, true, false, 3);
70              addBox.PackStart(helpButton, true, false, 3);
71              var doneButton = new Button("Done");
72              doneButton.Clicked += new EventHandler (OnDoneClick);
73              var exitButton = new Button("Exit");
74              exitButton.Clicked += new EventHandler(delegate{
75                  Application.Quit();
76              });
77
78              var optionsbox = new HBox(homogeneous: false, spacing: 3);
79              var optionbox1 = new VBox(homogeneous: false, spacing: 3);
80              var optionbox2 = new VBox(homogeneous: false, spacing: 3);
81              var optionbox3 = new VBox(homogeneous: false, spacing: 3);
82              optionbox1.PackStart(l1, true, true, 3);
83              optionbox1.PackStart(TimestepScale, true, true, 3);
84              optionbox2.PackStart(l2, true, true, 3);
85              optionbox2.PackStart(RScale, true, true, 3);
86              optionbox3.PackStart(l3, true, true, 3);
87              optionbox3.PackStart(LineScale, true, true, 3);
88              optionsbox.PackStart(optionbox1, true, true, 3);
89              optionsbox.PackStart(optionbox2, true, true, 3);
90              optionsbox.PackStart(optionbox3, true, true, 3);
91
92              controlbox.PackStart(optionsbox, false, false, 3);
93              controlbox.PackStart(addButton, false, false, 3);
94              controlbox.PackStart(addBox, false, false, 3);
95              systemscrollbox.Add(systembox);
96              donebox.PackStart(doneButton, true, true, 3);
97              donebox.PackStart(exitButton, true, true, 3);
98
99
100             containerbox.PackStart(controlbox, false, false, 3);
101             containerbox.PackStart(systemscrollbox, true, true, 3);
102             containerbox.PackStart(donebox, false, false, 3);
103             this.Add(containerbox);
104             this.ShowAll();
105         }
106         protected void OnDoneClick(object obj, EventArgs args) {
107             if (new_bodies.Count < 2) {
108                 Message("An empty system is not very interesting!");
109                 return;
110             }
111             try {
112                 Program.Program.CustomBodies.Clear();
113                 Program.Program.CustomCenters.Clear();
114                 centers.Clear();
115                 foreach (BodyBox b in new_bodies) {
116                     b.Set();
117                     Program.Program.CustomBodies.Add(b.body);
118                     centers.Add(b.CenterButton.Active);
119
120                 }
121                 Program.Program.CustomCenters = centers;
122                 Program.Program.radius_multiplier = RScale.Value;
123                 Program.Program.line_max = (int)LineScale.Value;
124                 Program.Program.timestep = TimestepScale.Value;
125                 Program.Program.StartSimulation();
126                 this.Destroy();
127             } catch (Exception e) {
128                 Message("I'm sorry, something went wrong but I don't know
    what. \nIf you can find a bored developer, show him this stack trace:\n" +
    e.Message + e.StackTrace);
129             }
130         }
```

```
131         protected void OnAddClick(object obj, EventArgs args) {
132
133             var bodyBox = new BodyBox(menu: this, homogeneous: false,
    spacing: 3);
134             String bString = BodyCombo.ActiveText;
135             if (bString != "Custom") {
136                 var body = Examples.solar_system.First(b => b.name ==
    bString);
137                 if (!(body.parent == null || new_bodies.Exists(b =>
    b.name.Text == body.parent.name))) {
138                     body = Examples.solar_system_bodies.First(b => b.name ==
    bString);
139                 }
140                 bodyBox.body = body;
141                 bodyBox.ReverseSet();
142             }
143
144             bodyBox.name.Text = BodyCombo.ActiveText;
145             systembox.PackStart(bodyBox, true, true, 3);
146             new_bodies.Add(bodyBox);
147             foreach (BodyBox b in new_bodies) {
148                 b.ResetParents();
149             }
150             this.ShowAll();
151         }
152         protected void OnSaveClick(object obj, EventArgs args) {
153             if (filename.Text == "") {
154                 Message("Please enter a filename");
155                 return;
156             }
157             System.Xml.Serialization.XmlSerializer writer =
158                 new System.Xml.Serialization.XmlSerializer(typeof(SaveData));
159             if (File.Exists(Environment.CurrentDirectory + "//" +
    filename.Text + ".xml")) {
160                 File.Delete(Environment.CurrentDirectory + "//" +
    filename.Text + ".xml");
161             }
162             FileStream file = File.Create(
163                 Environment.CurrentDirectory + "//" + filename.Text + ".xml");
164             var bodies = new List<Body>();
165             if (centers == null) centers = new List<bool>();
166             centers.Clear();
167             var elements = new List<OrbitalElements>();
168             foreach (BodyBox b in new_bodies) {
169                 b.Set();
170                 bodies.Add(b.body);
171                 centers.Add(b.CenterButton.Active);
172                 elements.Add(new OrbitalElements() {
173                     semilatusrectum = b.SLRScale.Value*AU,
174                     eccentricity = b.EScale.Value,
175                     inclination = b.IncScale.Value*deg,
176                     ascendingNodeLongitude = b.ANLScale.Value*deg,
177                     periapsisArgument = b.PAScale.Value*deg,
178                     trueAnomaly = b.TAScale.Value*deg
179                 });
180             }
181             var data = new SaveData() {
182                 bodies = bodies,
183                 elements = elements,
184                 timestep = TimestepScale.Value,
185                 centers = centers,
186                 radius_multiplier = RScale.Value,
187                 line_max = LineScale.Value,
188             };
189             writer.Serialize(file, data);
190             file.Close();
```

```
191         }
192         protected void OnLoadClick(object obj, EventArgs args) {
193             System.Xml.Serialization.XmlSerializer reader =
194                 new System.Xml.Serialization.XmlSerializer(typeof(SaveData));
195             SaveData data = new SaveData(); // To prevent compiler error
196             if (temp_savedata != null) {
197                 data = temp_savedata;
198                 temp_savedata = null;
199             } else {
200                 try {
201                     var file = new StreamReader(Environment.CurrentDirectory
    + "//" + filename.Text + ".xml");
202                     data = (SaveData)reader.Deserialize(file);
203                 } catch (IOException) {
204                     // Try in the system directory
205                     try {
206                         var file = new StreamReader
    (Environment.CurrentDirectory + "//" + SYSTEM_DIRECTORY + "//" +
    filename.Text + ".xml");
207                         data = (SaveData)reader.Deserialize(file);
208                     } catch (IOException) {
209                         Message("The specified file could not be found. Check
    that the name is spelt correctly and that it is in the correct directory");
210                         // cannot deserialize, exit
211                         return;
212                     }
213                 } catch (InvalidOperationException) {
214                     Message("The file is not a valid save file of this
    project");
215                     // cannot deserialize, exit
216                     return;
217                 }
218             }
219             RScale.Value = data.radius_multiplier;
220             LineScale.Value = data.line_max;
221             TimestepScale.Value = data.timestep;
222             new_bodies.Clear();
223             foreach (Widget w in systembox.Children) {
224                 if (w is BodyBox) systembox.Remove (w);
225             }
226             for (int i = 0; i < data.bodies.Count; i++) {
227                 var bbox = new BodyBox(menu: this, homogeneous: false,
    spacing: 3) {
228                     body = data.bodies[i],
229                 };
230                 bbox.CenterButton.Active = data.centers[i];
231                 if (data.elements != null && data.elements.Count != 0) {
232                     bbox.SetElements(data.elements[i]);
233                     bbox.ReverseSet(false);
234                 } else bbox.ReverseSet();
235                 new_bodies.Add(bbox);
236                 systembox.PackStart(bbox, true, true, 3);
237             }
238             foreach (BodyBox b in new_bodies) {
239                 b.ResetParents();
240             }
241             this.ShowAll();
242         }
243         protected void OnHelpClick(object obj, System.EventArgs args) {
244            OpenHTML("help.html");
245         }
246         protected void Message(String s) {
247             var window = new Window("Message");
248             var container = new VBox(homogeneous: true, spacing: 3);
249             window.Add(container);
250             container.PackStart(new Label(s), false, false, 3);
```

```
251             var closeButton = new Button("Close");
252             closeButton.Clicked += delegate { window.Destroy(); };
253             container.PackStart(closeButton, false, false, 3);
254             window.ShowAll();
255         }
256         protected void OpenHTML(String relPath) {
257             System.Threading.Tasks.Task.Run(() =>
    System.Diagnostics.Process.Start(relPath));
258         }
259         public void Remove(BodyBox b) {
260             var name = b.name.Text;
261             new_bodies.Remove(b);
262             systembox.Remove(b);
263             foreach (BodyBox a in new_bodies) {
264                 a.ResetParents();
265 
266             }
267         }
268         public void OnNameChanged(object obj, EventArgs args) {
269             foreach (BodyBox b in new_bodies) {
270                 b.ResetParents();
271             }
272         }
273     }
274     public class BodyBox : HBox {
275         public Body body {get; set;}
276         public Entry name {get; set;}
277         public ComboBoxText parent {get; set;} = new ComboBoxText();
278         public Scale MassScale {get; set;}
279         public Scale RadiusScale {get; set;}
280         public Scale SLRScale {get; set;}
281         public Scale EScale {get; set;}
282         public Scale IncScale {get; set;}
283         public Scale ANLScale {get; set;}
284         public Scale PAScale {get; set;}
285         public Scale TAScale {get; set;}
286         public Scale RScale {get; set;}
287         public Scale GScale {get; set;}
288         public Scale BScale {get; set;}
289         public CheckButton CenterButton {get; set;}
290         public Button DeleteButton {get; set;}
291         private static readonly double ECCENTRICITY_MAX = 3;
292         public BodyBox() {}
293         public BodyBox(Menu menu, bool homogeneous = false, int spacing =
    3) : base(homogeneous,spacing) {
294             body = new Structures.Body();
295             name = new Entry();
296             name.IsEditable = true;
297             name.Changed += new EventHandler(menu.OnNameChanged);
298             ResetParents();
299             MassScale = new Scale(Orientation.Vertical, 0.1,50,0.01);
300             RadiusScale = new Scale(Orientation.Vertical, 0.1,1000000,0.1);
301             SLRScale = new Scale(Orientation.Vertical, 0.1,50,0.01);
302             EScale = new Scale(Orientation.Vertical,
    0,ECCENTRICITY_MAX,0.001);
303             IncScale = new Scale(Orientation.Vertical, 0,180,0.01);
304             ANLScale = new Scale(Orientation.Vertical, 0,359.99,0.01);
305             PAScale = new Scale(Orientation.Vertical, 0,359.99,0.01);
306             TAScale = new Scale(Orientation.Vertical, 0,359.99,0.01);
307             RScale = new Scale(Orientation.Horizontal, 0, 1, 0.01);
308             RScale.Value = 1;
309             GScale = new Scale(Orientation.Horizontal, 0, 1, 0.01);
310             GScale.Value = 1;
311             BScale = new Scale(Orientation.Horizontal, 0, 1, 0.01);
312             BScale.Value = 1;
313             CenterButton = new CheckButton("Focusable");
```

```
314             DeleteButton = new Button("Delete");
315             DeleteButton.Clicked += new EventHandler(OnDeleteClick);
316 
317             parent.Changed += new EventHandler(OnParentChange);
318             MassScale.Inverted = true;
319             RadiusScale.Inverted = true;
320             SLRScale.Inverted = true;
321             EScale.Inverted = true;
322             IncScale.Inverted = true;
323             ANLScale.Inverted = true;
324             PAScale.Inverted = true;
325             TAScale.Inverted = true;
326             var pBox = new VBox(homogeneous: false, spacing: 3);
327             pBox.PackStart(new Label("Parent Body"), false, false, 3);
    pBox.PackStart(parent, true, true, 3);
328             var mBox = new VBox(homogeneous: false, spacing: 3);
329             mBox.PackStart(new Label("ln(m)"), false, false, 3);
    mBox.PackStart(MassScale, true, true, 3);
330             var rBox = new VBox(homogeneous: false, spacing: 3);
331             rBox.PackStart(new Label("r (km)"), false, false, 3);
    rBox.PackStart(RadiusScale, true, true, 3);
332             var slrBox = new VBox(homogeneous: false, spacing: 3);
333             slrBox.PackStart(new Label("ρ (AU)"), false, false, 3);
    slrBox.PackStart(SLRScale, true, true, 3);
334             var eBox = new VBox(homogeneous: false, spacing: 3);
335             eBox.PackStart(new Label("e"), false, false, 3); eBox.PackStart
    (EScale, true, true, 3);
336             var incBox = new VBox(homogeneous: false, spacing: 3);
337             incBox.PackStart(new Label("i (°)"), false, false, 3);
    incBox.PackStart(IncScale, true, true, 3);
338             var anlBox = new VBox(homogeneous: false, spacing: 3);
339             anlBox.PackStart(new Label("Ω (°)"), false, false, 3);
    anlBox.PackStart(ANLScale, true, true, 3);
340             var paBox = new VBox(homogeneous: false, spacing: 3);
341             paBox.PackStart(new Label("ω (°)"), false, false, 3);
    paBox.PackStart(PAScale, true, true, 3);
342             var taBox = new VBox(homogeneous: false, spacing: 3);
343             taBox.PackStart(new Label("ν (°)"), false, false, 3);
    taBox.PackStart(TAScale, true, true, 3);
344 
345             this.PackStart(name, true, true, 3);
346             this.PackStart(pBox, false, false, 3);
347             this.PackStart(mBox, true, true, 3);
348             this.PackStart(rBox, true, true, 3);
349             this.PackStart(slrBox, true, true, 3);
350             this.PackStart(eBox, true, true, 3);
351             this.PackStart(incBox, true, true, 3);
352             this.PackStart(anlBox, true, true, 3);
353             this.PackStart(paBox, true, true, 3);
354             this.PackStart(taBox, true, true, 3);
355 
356             var colorbox = new VBox(homogeneous: false, spacing: 3);
357             colorbox.PackStart(new Label("RGB"), false, false, 3);
358             colorbox.PackStart(RScale, true, true, 3);
359             colorbox.PackStart(GScale, true, true, 3);
360             colorbox.PackStart(BScale, true, true, 3);
361 
362             this.PackStart(colorbox, true, true, 3);
363             var optionsbox = new VBox(homogeneous: false, spacing: 3);
364             optionsbox.PackStart(CenterButton, true, true, 3);
365             optionsbox.PackStart(DeleteButton, true, true, 3);
366             this.PackStart(optionsbox, true, true, 3);
367 
368         }
369         protected void OnParentChange(object obj, EventArgs args) {
370             try {
```

```
371                 var parentBody = menu.new_bodies.FirstOrDefault(b =>
    b.body.name == parent.ActiveText).body;
372                 double hillrad = parentBody.HillRadius()/AU;
373                 this.SLRScale.Digits = Math.Max(0,8);//3-(int)Math.Log
    (hillrad/100000));
374                 this.SLRScale.SetIncrements(Math.Pow(10,-
    this.SLRScale.Digits),hillrad/100000);
375                 this.SLRScale.SetRange(Math.Pow(10,-
    this.SLRScale.Digits),hillrad);
376             } catch (NullReferenceException) {} // no parent, don't set values
377         }
378         protected void OnDeleteClick(object obj, EventArgs args) {
379             menu.Remove(this);
380             menu.ShowAll();
381             this.Destroy();
382         }
383         public void Set() {
384             if (parent.ActiveText != this.name.Text && parent.Active != -1) {
385                 var elements = new Structures.OrbitalElements() {
386                     semilatusrectum = SLRScale.Value*AU,
387                     eccentricity = EScale.Value,
388                     inclination = IncScale.Value*deg,
389                     ascendingNodeLongitude = ANLScale.Value*deg,
390                     periapsisArgument = PAScale.Value*deg,
391                     trueAnomaly = TAScale.Value*deg
392                 };
393                 body = new Structures.Body(menu.new_bodies.FirstOrDefault(b
    => b.body.name == parent.ActiveText).body,elements);
394             }
395             body.name = this.name.Text;
396             body.stdGrav = Math.Pow(Math.E,MassScale.Value)*G*1e22;
397             body.radius = RadiusScale.Value*1e3;
398             body.color = new Vector3(RScale.Value, GScale.Value,
    BScale.Value);
399         }
400         public void SetElements(OrbitalElements elements) {
401             try {
402                 if (elements.semilatusrectum > this.body.parent.HillRadius()/
    AU) {
403                     SLRScale.SetRange(1e-8,elements.semilatusrectum);
404                 }
405             } catch (NullReferenceException) {} // body has no parent, we
    cannot check the slr
406             SLRScale.Value = elements.semilatusrectum/AU;
407             if (elements.eccentricity > ECCENTRICITY_MAX) {
408                 EScale.SetRange(0,elements.eccentricity);
409             } else {
410                 EScale.SetRange(0, ECCENTRICITY_MAX); // there is no way to
    see the current range, so we'll set it every time
411             }
412             EScale.Value   = elements.eccentricity;
413             IncScale.Value = elements.inclination/deg;
414             ANLScale.Value = elements.ascendingNodeLongitude/deg;
415             PAScale.Value  = elements.periapsisArgument/deg;
416             TAScale.Value  = elements.trueAnomaly/deg;
417         }
418         public void ReverseSet(bool elem = true) {
419             if (elem) try {
420                 parent.Active  = menu.new_bodies.FindIndex(b => b.name.Text
    == body.parent.name);
421                 var elements   = new OrbitalElements(body.position-
    body.parent.position,body.velocity-body.parent.velocity,body.parent.stdGrav);
422                 this.SetElements(elements);
423             } catch (NullReferenceException) {} // if body has no parent
424             name.Text = body.name;
425             MassScale.Value = Math.Log((body.stdGrav/G)/1e22);
```

```
426                 RadiusScale.Value = body.radius/1e3;
427                 RScale.Value = body.color.x;
428                 GScale.Value = body.color.y;
429                 BScale.Value = body.color.z;
430             }
431             public void ResetParents() {
432                 parent.RemoveAll();
433                 foreach (BodyBox b in menu.new_bodies) {
434                     parent.AppendText(b.name.Text);
435                 }
436                 try {
437                     parent.Active = menu.new_bodies.FindIndex(b => b.name.Text ==
    body.parent.name);
438                 } catch (NullReferenceException) {} // parent no longer exists
439 
440             }
441         }
442         [Serializable()]
443         public class SaveData {
444             public List<Body> bodies {get; set;}
445             public List<OrbitalElements> elements {get; set;}
446             public List<bool> centers {get; set;}
447             public double timestep {get; set;}
448             public double radius_multiplier {get; set;}
449             public double line_max {get; set;}
450         }
451     }
```