```
1   /*************************************************************************
2    * Author       : Jessica Sulzbach
3    * Class        : I.In-P4B
4    * Project      : TPI - Virtual animal
5    * Name         : VirtualAnimalView
6    * Description  : Main form
7    * Last modified: 23.05.2017
8    *************************************************************************/
9   using System;
10  using System.Collections.Generic;
11  using System.ComponentModel;
12  using System.Data;
13  using System.Drawing;
14  using System.Linq;
15  using System.Text;
16  using System.Threading;
17  using System.Threading.Tasks;
18  using System.Windows.Forms;
19
20  namespace VirtualAnimal
21  {
22      public partial class VirtualAnimalView : Form
23      {
24
25          #region Variables
26          private Animal _theAnimal;
27          private DataRecovery _saveOrRecover;
28          private Inventory _theInventory;
29          // Background image
30          private Image background;
31          // Animation variables (time and animation name)
32          private int time;
33          private string animName;
34
35          #endregion
36
37          #region Properties
38          internal DataRecovery SaveOrRecover
39          {
40              get { return _saveOrRecover; }
41              set { _saveOrRecover = value; }
42          }
43          internal Animal TheAnimal
44          {
45              get { return _theAnimal; }
46              set { _theAnimal = value; }
47          }
48
49          internal Inventory TheInventory
50          {
51              get { return _theInventory; }
52              set { _theInventory = value; }
53          }
54
55          public Image Background
56          {
57              get { return background; }
```

```
58              set { background = value; }
59          }
60          #endregion
61
62          #region Constructor
63          //Constructor
64          public VirtualAnimalView()
65          {
66              InitializeComponent();
67
68              DoubleBuffered = true;
69
70              // Transparent background for the animal pictureBox
71              pbxAnimalAnimation.BackColor = Color.Transparent;
72              // Set the home background image
73              Background = Properties.Resources.BackgroundHome;
74
75              // Iniciate class
76              TheAnimal = new Animal();
77              SaveOrRecover = new DataRecovery();
78              TheInventory = new Inventory();
79
80              // Reset the animation variables
81              time = 0;
82              TheAnimal.NumImage = 0;
83              animName = "";
84              TheInventory.Product = "";
85          }
86          #endregion
87
88          private void VirtualAnimalView_Load(object sender, EventArgs e)
89          {
90              // If new game
91              if (TheAnimal.Age == 0)
92              {
93                  time = 0;
94                  TheInventory.RewriteNew();
95                  TheAnimal.NumImage = 0;
96                  animName = "Born";
97                  TheAnimal.Animations(animName);
98                  pbxAnimalAnimation.Location = new Point(234, 114);
99                  gbxProgressBar.Visible = false;
100                 pbxSmileyFace.Visible = false;
101                 btnAnimal.Visible = false;
102                 btnGoOut.Visible = false;
103                 btnInventory.Visible = false;
104             }
105             // If continuing game
106             else
107             {
108                 pbxAnimalAnimation.Location = new Point(352, 187);
109                 animName = "Idle";
110                 TheAnimal.Animations(animName);
111             }
112
113                 // Iniciate the necessary components for ProgressBar and Animations
                    (value and timer)
```

```
114                 AnimationsInitialize();
115                 ProgressBarInitialize();
116 
117             }
118 
119             #region ProgressBar
120             /// <summary>
121             /// Iniciates the progressBars value and timer
122             /// </summary>
123             private void ProgressBarInitialize()
124             {
125                 // Iniciates the progressBars value
126                 UpdateProgressBar();
127 
128                 // Iniciates the progressBars timer
129                 tmrProgressBar.Enabled = true;
130                 tmrProgressBar.Start();
131                 tmrProgressBar.Interval = 1000;
132             }
133 
134             // If the value is higher then 0, then minus 1
135             private void tmrProgressBar_Tick(object sender, EventArgs e)
136             {
137                 if (TheAnimal.Health > 0)
138                 {
139                     TheAnimal.Health--;
140                 }
141                 if (TheAnimal.Hygene > 0)
142                 {
143                     TheAnimal.Hygene--;
144                 }
145 
146                 if (TheAnimal.Energy > 0)
147                 {
148                     TheAnimal.Energy--;
149                 }
150                 if (TheAnimal.Happiness > 0)
151                 {
152                     TheAnimal.Happiness--;
153                 }
154 
155                 // If one of the life levels equal 0 or less -> GAME OVER
156                 if (TheAnimal.Happiness <= 0 || TheAnimal.Health <= 0 || TheAnimal.Hygene
                     <= 0 || TheAnimal.Energy <= 0)
157                 {
158                     animName = "Death";
159                     TheAnimal.NumImage = 0;
160                     time = 0;
161                     TheAnimal.Animations(animName);
162                     TheInventory.Product = "";
163                     tmrProgressBar.Stop();
164                 }
165 
166                 // According to life levels the smiley image will change
167                 // Condition : The life levels have to be bigger than 50 and smaller
                    then 69
168                 if ((TheAnimal.Happiness > 50 && TheAnimal.Happiness < 69) || (TheAnimal.
```

```
            Health > 50 && TheAnimal.Health < 69) || (TheAnimal.Hygene > 50 &&
            TheAnimal.Hygene < 69) || (TheAnimal.Energy > 50 && TheAnimal.Energy < 69
            ))
169         {
170             pbxSmileyFace.Image = Properties.Resources.MiddleFace;
171         }
172         // Condition : The life levels have to smaller than 20 or equal 20
173         else if (TheAnimal.Happiness <= 20 || TheAnimal.Health <= 20 || TheAnimal
            .Hygene <= 20 || TheAnimal.Energy <= 20)
174         {
175             pbxSmileyFace.Image = Properties.Resources.SadFace;
176         }
177         // Condition : The life levels are high
178         else
179         {
180             pbxSmileyFace.Image = Properties.Resources.HappyFace;
181         }
182
183         // Counts the aniamls age by seconde
184         TheAnimal.Age++;
185
186         // Updates the progressBar life levels
187         UpdateProgressBar();
188     }
189
190     /// <summary>
191     /// Updates the progressBar life levels with the life level values
192     /// </summary>
193     private void UpdateProgressBar()
194     {
195         prbEnergy.Value = TheAnimal.Energy;
196
197         prbHappiness.Value = TheAnimal.Happiness;
198
199         prbHealth.Value = TheAnimal.Health;
200
201         prbHygene.Value = TheAnimal.Hygene;
202     }
203
204     #endregion
205
206     #region DropDownButtons
207     private void btnAnimal_Click(object sender, EventArgs e)
208     {
209         // Opens the contextMenuStrip under the button, giving the illusion of a
            dropdown button
210         cmsForAnimalButton.Show(btnAnimal, new Point(0, btnAnimal.Height));
211     }
212
213     private void btnInventory_Click(object sender, EventArgs e)
214     {
215         // Opens the contextMenuStrip under the button, giving the illusion of a
            dropdown button
216         cmsForInventoryButton.Show(btnInventory, new Point(0, btnInventory.Height
            ));
217     }
218
```

```
219         private void btnGoOut_Click(object sender, EventArgs e)
220         {
221             // Opens the contextMenuStrip under the button, giving the illusion of a
                dropdown button
222             cmsForGoOutButton.Show(btnGoOut, new Point(0, btnGoOut.Height));
223         }
224 
225         /// <summary>
226         /// Prepares the images and updates the life levels for animation HAPPY
227         /// </summary>
228         /// <param name="sender"></param>
229         /// <param name="e"></param>
230         private void tsmPet_Click(object sender, EventArgs e)
231         {
232             TheAnimal.NumImage = 0;
233             animName = "Happy";
234             TheAnimal.Animations("Happy");
235             UpdateProgressBar();
236         }
237 
238         /// <summary>
239         /// Prepares the images and updates the life levels for animation SLEEP
240         /// </summary>
241         /// <param name="sender"></param>
242         /// <param name="e"></param>
243         private void tsmSleep_Click(object sender, EventArgs e)
244         {
245             TheAnimal.NumImage = 0;
246             animName = "Sleep";
247             TheAnimal.Animations(animName);
248             UpdateProgressBar();
249 
250         }
251 
252         /// <summary>
253         /// Prepares the images for animation SLEEP and updates the life levels for
            NAP
254         /// </summary>
255         /// <param name="sender"></param>
256         /// <param name="e"></param>
257         private void tsmNap_Click(object sender, EventArgs e)
258         {
259             TheAnimal.NumImage = 0;
260             animName = "Sleep";
261             TheAnimal.Animations("Nap");
262             UpdateProgressBar();
263         }
264 
265         /// <summary>
266         /// Prepares the images and updates the life levels for animation Walk
267         /// </summary>
268         /// <param name="sender"></param>
269         /// <param name="e"></param>
270         private void tsmWalk_Click(object sender, EventArgs e)
271         {
272             TheAnimal.NumImage = 0;
273             animName = "Walk";
```

```
274             TheInventory.Product = "";
275             TheAnimal.Animations(animName);
276             UpdateProgressBar();
277         }
278         #endregion
279
280         #region OpenForms
281         // Opens the store view as the child of VirtualAnimalView
282         private void tsmStore_Click(object sender, EventArgs e)
283         {
284             VirtualAnimalStore store = new VirtualAnimalStore();
285             store.ShowDialog(this);
286         }
287
288         // Opens the food view as the child of VirtualAnimalView
289         private void tsmFood_Click(object sender, EventArgs e)
290         {
291             VirtualAnimalFood food = new VirtualAnimalFood(TheInventory);
292             food.ShowDialog(this);
293
294             // Iniciates the images and updates progressBar according to the Product
                use
295             TheAnimal.Animations(TheInventory.Product);
296             UpdateProgressBar();
297         }
298
299         // Opens the materials view as the child of VirtualAnimalView
300         private void tsmMaterials_Click(object sender, EventArgs e)
301         {
302             VirtualAnimalMaterials materials = new VirtualAnimalMaterials(
                TheInventory);
303             materials.ShowDialog(this);
304
305             // Iniciates the images and updates progressBar according to the Product
                use
306             TheAnimal.Animations(TheInventory.Product);
307             UpdateProgressBar();
308         }
309
310         #endregion
311
312         #region Animations
313         private void AnimationsInitialize()
314         {
315             // Iniciates the animation timer
316             tmrAnimalAnimations.Enabled = true;
317             tmrAnimalAnimations.Start();
318             tmrAnimalAnimations.Interval = 150;
319         }
320
321         /// <summary>
322         /// Updates the image in the pictureBox
323         /// </summary>
324         /// <param name="numImage">The image number</param>
325         private void updateAnim(int numImage)
326         {
327             this.pbxAnimalAnimation.Image = TheAnimal.Anim[numImage];
```

```
328         }
329
330         /// <summary>
331         /// Every tick it checks for conditions. If conditions are true then a
            animation will start.
332         /// </summary>
333         /// <param name="sender"></param>
334         /// <param name="e"></param>
335         private void tmrAnimalAnimations_Tick(object sender, EventArgs e)
336         {
337             if (animName == "Happy" || TheInventory.Product == "Happy")
338             {
339                 if (time <= 8)
340                 {
341                     if (TheAnimal.NumImage <= 8)
342                     {
343                         updateAnim(TheAnimal.NumImage);
344                         time++;
345                         TheAnimal.NumImage++;
346                     }
347                 }
348                 else
349                 {
350                     time = 0;
351                     TheAnimal.Animations("Idle");
352                     animName = "Idle";
353                     TheInventory.Product = "Idle";
354                 }
355             }
356             else if (animName == "Sleep" || animName == "Nap")
357             {
358                 if (time <= 18)
359                 {
360                     if (TheAnimal.NumImage <= 18)
361                     {
362                         updateAnim(TheAnimal.NumImage);
363                         time++;
364                         TheAnimal.NumImage++;
365                     }
366                 }
367                 else
368                 {
369                     time = 0;
370                     TheAnimal.Animations("Idle");
371                     animName = "Idle";
372                 }
373             }
374             else if (animName == "Walk")
375             {
376                 Background = Properties.Resources.WalkBackground;
377                 lblGift.Visible = false;
378                 if (time <= 25)
379                 {
380                     if (TheAnimal.Gifts.ContainsKey(time))
381                     {
382                         lblGift.Text = " + " + Convert.ToString(TheAnimal.Gifts[time
                            ]);
```

```
383                         lblGift.Visible = true;
384                         TheAnimal.Money += TheAnimal.Gifts[time];
385                     }
386                     if (time == 0)
387                     {
388                         pbxAnimalAnimation.Location = new Point(218, 165);
389                     }
390                     if (TheAnimal.NumImage == 0)
391                     {
392                         TheAnimal.NumImage = 1;
393                         updateAnim(TheAnimal.NumImage);
394                     }
395                     else
396                     {
397                         TheAnimal.NumImage = 0;
398                         updateAnim(TheAnimal.NumImage);
399                     }
400                     time++;
401                     pbxAnimalAnimation.Left += 10;
402                 }
403                 else
404                 {
405                     time = 0;
406                     Background = Properties.Resources.BackgroundHome;
407                     pbxAnimalAnimation.Location = new Point(352, 187);
408                     TheAnimal.Animations("Idle");
409                     animName = "Idle";
410                 }
411                 this.Refresh();
412             }
413             else if (TheInventory.Product == "Eat")
414             {
415                 if (time <= 10)
416                 {
417                     if (TheAnimal.NumImage == 0)
418                     {
419                         TheAnimal.NumImage = 1;
420                         updateAnim(TheAnimal.NumImage);
421                         time++;
422                     }
423                     else
424                     {
425                         TheAnimal.NumImage = 0;
426                         updateAnim(TheAnimal.NumImage);
427                         time++;
428                     }
429                 }
430                 else
431                 {
432                     time = 0;
433                     TheAnimal.Animations("Idle");
434                     TheInventory.Product = "Idle";
435                 }
436             }
437             else if (TheInventory.Product == "Shower")
438             {
439                 Background = Properties.Resources.ShowerBackground;
```

```
440                 if (time <= 14)
441                 {
442                     if (TheAnimal.NumImage <= 7)
443                     {
444                         updateAnim(TheAnimal.NumImage);
445                         time++;
446                         TheAnimal.NumImage++;
447                     }
448                     else
449                     {
450                         TheAnimal.NumImage = 0;
451                     }
452                 }
453                 else
454                 {
455                     time = 0;
456                     Background = Properties.Resources.BackgroundHome;
457                     TheAnimal.Animations("Idle");
458                     TheInventory.Product = "Idle";
459 
460                 }
461                 this.Refresh();
462             }
463             else if (TheInventory.Product == "Brush")
464             {
465                 if (time <= 10)
466                 {
467                     if (TheAnimal.NumImage == 0)
468                     {
469                         TheAnimal.NumImage = 1;
470                         updateAnim(TheAnimal.NumImage);
471                         time++;
472                     }
473                     else
474                     {
475                         TheAnimal.NumImage = 0;
476                         updateAnim(TheAnimal.NumImage);
477                         time++;
478                     }
479                 }
480                 else
481                 {
482                     time = 0;
483                     TheAnimal.Animations("Idle");
484                     TheInventory.Product = "Idle";
485                 }
486             }
487             else if (animName == "Idle" || TheInventory.Product == "Idle")
488             {
489                 if (TheAnimal.NumImage <= 1)
490                 {
491                     updateAnim(TheAnimal.NumImage);
492                     TheAnimal.NumImage++;
493                 }
494                 else
495                 {
496                     updateAnim(TheAnimal.NumImage);
```

```
497                     TheAnimal.NumImage = 0;
498                 }
499 
500             }
501             else if (animName == "Death")
502             {
503                 if (time <= 4)
504                 {
505                     if (TheAnimal.NumImage <= 4)
506                     {
507                         updateAnim(TheAnimal.NumImage);
508                         time++;
509                         TheAnimal.NumImage++;
510                     }
511                 }
512                 else
513                 {
514                     animName = "";
515                     string playerClass = "";
516                     if(this.TheAnimal.Age <= 599)
517                     {
518                         playerClass = "Il faut fair plus d'effort!";
519                     }
520                     else if(this.TheAnimal.Age >= 600 && this.TheAnimal.Age <=1099)
521                     {
522                         playerClass = "Il y a une marge d'améliorations!";
523                     }
524                     else if (this.TheAnimal.Age >= 1100 && this.TheAnimal.Age <= 1699)
525                     {
526                         playerClass = "Bien jouer!";
527                     }
528                     else if (this.TheAnimal.Age >= 1700 && this.TheAnimal.Age <= 2299)
529                     {
530                         playerClass = "Excellent!";
531                     }
532                     else if (this.TheAnimal.Age >= 2300)
533                     {
534                         playerClass = "Incroyable!!";
535                     }
536 
537                     MessageBox.Show("R.I.P " + this.TheAnimal.Name+ "..." +
                        playerClass, "Game over", MessageBoxButtons.OK,MessageBoxIcon.
                        Exclamation);
538                     time = 0;
539                     TheAnimal.Age = 0;
540                     TheInventory.RewriteNew();
541                     TheAnimal.NumImage = 0;
542                     animName = "Born";
543                     TheAnimal.Animations(animName);
544                     pbxAnimalAnimation.Location = new Point(234, 114);
545                     gbxProgressBar.Visible = false;
546                     pbxSmileyFace.Visible = false;
547                     btnAnimal.Visible = false;
548                     btnGoOut.Visible = false;
549                     btnInventory.Visible = false;
550                 }
551             }
```

```
552             else if (animName == "Born")
553             {
554                 Background = Properties.Resources.background;
555                 if (time <= 8)
556                 {
557                     if (TheAnimal.NumImage <= 8)
558                     {
559                         updateAnim(TheAnimal.NumImage);
560                         time++;
561                         TheAnimal.NumImage++;
562                     }
563                 }
564                 else
565                 {
566                     lblName.Visible = true;
567                     tbxName.Visible = true;
568                     btnStart.Visible = true;
569                 }
570                 this.Refresh();
571             }
572         }
573 
574         private void VirtualAnimalView_Paint(object sender, PaintEventArgs e)
575         {
576             e.Graphics.DrawImage(Background, 218, 12, 350, 297);
577         }
578         #endregion
579 
580         private void VirtualAnimalView_FormClosing(object sender,
            FormClosingEventArgs e)
581         {
582 
583             tmrProgressBar.Enabled = false;
584 
585             TheAnimal.SaveOrRecover.FirstTime = true;
586             this.TheAnimal.AnimalSave(TheAnimal.Health);
587             this.TheAnimal.AnimalSave(TheAnimal.Hygene);
588             this.TheAnimal.AnimalSave(TheAnimal.Energy);
589             this.TheAnimal.AnimalSave(TheAnimal.Happiness);
590             this.TheAnimal.AnimalSave(Convert.ToInt32(TheAnimal.Money));
591 
592             TheAnimal.SaveOrRecover.FirstTime = true;
593             this.TheAnimal.SaveOrRecover.FileWritter("Animal_Age_Name.txt", TheAnimal
                .Name);
594             this.TheAnimal.SaveOrRecover.FileWritter("Animal_Age_Name.txt", Convert.
                ToString(TheAnimal.Age));
595         }
596 
597         private void btnStart_Click(object sender, EventArgs e)
598         {
599             TheAnimal.Name = tbxName.Text;
600             tbxName.Text = "";
601 
602             TheAnimal.SaveOrRecover.FirstTime = true;
603             TheAnimal.SaveOrRecover.FileWritter("Animal_Age_Name.txt", TheAnimal.Name
                );
604             TheAnimal.SaveOrRecover.FileWritter("Animal_Age_Name.txt", Convert.
```

```
                ToString(0));
605
606             lblName.Visible = false;
607             tbxName.Visible = false;
608             btnStart.Visible = false;
609             gbxProgressBar.Visible = true;
610             pbxSmileyFace.Visible = true;
611             btnAnimal.Visible = true;
612             btnGoOut.Visible = true;
613             btnInventory.Visible = true;
614
615             background = Properties.Resources.BackgroundHome;
616             pbxAnimalAnimation.Location = new Point(353, 165);
617             this.Refresh();
618             animName = "Idle";
619             TheAnimal.Animations(animName);
620             AnimationsInitialize();
621             ProgressBarInitialize();
622         }
623     }
624 }
```

```
/*************************************************************************
 * Author       : Jessica Sulzbach
 * Class        : I.In-P4B
 * Project      : TPI - Virtual animal
 * Name         : VirtualAnimalFood
 * Description  : Food form
 * Last modified: 23.05.2017
 *************************************************************************/
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace VirtualAnimal
{
    public partial class VirtualAnimalFood : Form
    {
        // Variable
        private Inventory _theInventory;

        // Propertie
        public Inventory TheInventory
        {
            get { return _theInventory; }
            set { _theInventory = value; }
        }

        public VirtualAnimalFood(Inventory i)
        {
            InitializeComponent();
            TheInventory = i;

            TheInventory.InventoryData();
            TheInventory.InventoryData("Food");
        }

        private void VirtualAnimalFood_Load(object sender, EventArgs e)
        {
            UpdateView();
        }

        /// <summary>
        /// Fills the table panel layout with the labels and radio buttons necessary
        /// </summary>
        private void UpdateView()
        {
            TheInventory.SaveOrRecover.FileReader("Product_name_and_price.txt");
            TheInventory.InventoryData("Food");
            tlpFood.Controls.Clear();

            int Line = 1;

```

```
58              tlpFood.Controls.Add(new Label() { Text = "Produits", Anchor =
                AnchorStyles.None, AutoSize = true, Font = new Font("Verdana", 11,
                FontStyle.Bold) }, 0, 0);
59              tlpFood.Controls.Add(new Label() { Text = "Quantité", Anchor =
                AnchorStyles.None, AutoSize = true, Font = new Font("Verdana", 11,
                FontStyle.Bold) }, 1, 0);
60              foreach (var pair in TheInventory.DataInventoryHALF)
61              {
62                  tlpFood.Controls.Add(new RadioButton() { Name = "rdbFood" + Line,
                    Text = pair.Key, Anchor = AnchorStyles.Left, AutoSize = true, Font =
                    new Font("Verdana", 11, FontStyle.Regular) }, 0, Line);
63                  tlpFood.Controls.Add(new Label() { Name = "lblFood" + Line, Text =
                    string.Format("{0}", pair.Value), Anchor = AnchorStyles.None,
                    AutoSize = true, Font = new Font("Verdana", 11, FontStyle.Regular) },
                     1, Line);
64                  Line++;
65              }
66          }
67
68          /// <summary>
69          /// Quantity minus 1
70          /// Closes the form and a animation plays
71          /// </summary>
72          private void btnFoodUse_Click(object sender, EventArgs e)
73          {
74              string myKey;
75              int myValue;
76              for (int i = 1; i <= TheInventory.DataInventoryHALF.Count; i++)
77              {
78                  if (((RadioButton)tlpFood.Controls["rdbFood" + (i)]).Checked)
79                  {
80                      myKey = ((RadioButton)tlpFood.Controls["rdbFood" + (i)]).Text;
81                      myValue = Convert.ToInt32(((Label)tlpFood.Controls["lblFood" + (i
                        )]).Text);
82
83                      if (myValue > 0)
84                      {
85                          TheInventory.DataInventoryFULL[myKey] = myValue - 1;
86                          TheInventory.DataInventoryHALF[myKey] = myValue - 1;
87
88                          TheInventory.Rewrite();
89                          if (myKey == "Riz" || myKey == "Sushi")
90                          {
91                              TheInventory.Use("Eat");
92                          }
93                          else
94                          {
95                              TheInventory.Use("Happy");
96                          }
97
98                      }
99                      else
100                     {
101                         MessageBox.Show("Oups... Vous n'avez plus de " + myKey + ",
                            il faut aller en acheter au magasin.", "Erreur",
                            MessageBoxButtons.OK, MessageBoxIcon.Stop);
102                     }
```

```
103                 }
104             }
105             this.Close();
106         }
107 
108         /// <summary>
109         /// Form stays open.
110         /// The quantity equals 0
111         /// </summary>
112         private void btnFoodSell_Click(object sender, EventArgs e)
113         {
114             for (int i = 1; i <= TheInventory.DataInventoryHALF.Count; i++)
115             {
116                 if (((RadioButton)tlpFood.Controls["rdbFood" + (i)]).Checked)
117                 {
118                     if (TheInventory.DataInventoryHALF.ContainsKey(((RadioButton)
                        tlpFood.Controls["rdbFood" + (i)]).Text))
119                     {
120                         string myKey = ((RadioButton)tlpFood.Controls["rdbFood" + (i
                            )]).Text;
121 
122                         TheInventory.DataInventoryFULL[myKey] = 0;
123                         TheInventory.DataInventoryHALF[myKey] = 0;
124                     }
125                     TheInventory.Rewrite();
126                 }
127             }
128             UpdateView();
129         }
130 
131         private void btnFoodBack_Click(object sender, EventArgs e)
132         {
133             this.Close();
134         }
135     }
136 }
137 
```

```
1   /*************************************************************************
2    * Author       : Jessica Sulzbach
3    * Class        : I.In-P4B
4    * Project      : TPI - Virtual animal
5    * Name         : VirtualAnimalMaterials
6    * Description  : Materials form
7    * Last modified: 23.05.2017
8    *************************************************************************/
9   using System;
10  using System.Collections.Generic;
11  using System.ComponentModel;
12  using System.Data;
13  using System.Drawing;
14  using System.Linq;
15  using System.Text;
16  using System.Threading.Tasks;
17  using System.Windows.Forms;
18
19  namespace VirtualAnimal
20  {
21      public partial class VirtualAnimalMaterials : Form
22      {
23          // Variable
24          private Inventory _theInventory;
25
26          // Propertie
27          internal Inventory TheInventory
28          {
29              get { return _theInventory; }
30              set { _theInventory = value; }
31          }
32
33          public VirtualAnimalMaterials(Inventory i)
34          {
35              InitializeComponent();
36              TheInventory = i;
37
38              TheInventory.InventoryData();
39              TheInventory.InventoryData("Materials");
40          }
41
42          private void VirtualAnimalMaterials_Load(object sender, EventArgs e)
43          {
44              UpdateView();
45          }
46
47          /// <summary>
48          /// Fills the table panel layout with the labels and radio buttons necessary
49          /// </summary>
50          private void UpdateView()
51          {
52              TheInventory.SaveOrRecover.FileReader("Product_name_and_price.txt");
53              TheInventory.InventoryData("Materials");
54              tlpMaterials.Controls.Clear();
55
56              int Line = 1;
57
```

```
58              tlpMaterials.Controls.Add(new Label() { Text = "Produits", Anchor =
                AnchorStyles.None, AutoSize = true, Font = new Font("Verdana", 11,
                FontStyle.Bold) }, 0, 0);
59              tlpMaterials.Controls.Add(new Label() { Text = "Quantité", Anchor =
                AnchorStyles.None, AutoSize = true, Font = new Font("Verdana", 11,
                FontStyle.Bold) }, 1, 0);
60              foreach (var pair in TheInventory.DataInventoryHALF)
61              {
62                  tlpMaterials.Controls.Add(new RadioButton() { Name = "rdbMaterial" +
                    Line, Text = pair.Key, Anchor = AnchorStyles.Left, AutoSize = true,
                    Font = new Font("Verdana", 11, FontStyle.Regular) }, 0, Line);
63                  tlpMaterials.Controls.Add(new Label() { Name = "lblMaterial" + Line,
                    Text = string.Format("{0}", pair.Value), Anchor = AnchorStyles.None,
                    AutoSize = true, Font = new Font("Verdana", 11, FontStyle.Regular) },
                     1, Line);
64                  Line++;
65              }
66          }
67
68          /// <summary>
69          /// Quantity minus 1
70          /// Closes the form and a animation plays
71          /// </summary>
72          private void btnMaterialsUse_Click(object sender, EventArgs e)
73          {
74              string myKey;
75              int myValue;
76              for (int i = 1; i < TheInventory.DataInventoryHALF.Count; i++)
77              {
78                  if (((RadioButton)tlpMaterials.Controls["rdbMaterial" + (i)]).Checked)
79                  {
80                      myKey = ((RadioButton)tlpMaterials.Controls["rdbMaterial" + (i
                        )]).Text;
81                      myValue = Convert.ToInt32(((Label)tlpMaterials.Controls[
                        "lblMaterial" + (i)]).Text);
82
83                      if (myValue != 0)
84                      {
85                          TheInventory.DataInventoryFULL[myKey] = myValue - 1;
86                          TheInventory.DataInventoryHALF[myKey] = myValue - 1;
87
88                          TheInventory.Rewrite();
89                          if (myKey == "Shampooing")
90                          {
91                              TheInventory.Use("Shower");
92                          }
93                          else if(myKey == "Brosse")
94                          {
95                              TheInventory.Use("Brush");
96                          }
97                      }
98                      else
99                      {
100                         MessageBox.Show("Oups... Vous n'avez plus de " + myKey + ",
                            il faut aller en acheter au magasin.", "Erreur",
                            MessageBoxButtons.OK, MessageBoxIcon.Stop);
101                     }
```

```
102                 }
103             }
104             this.Close();
105         }
106 
107         /// <summary>
108         /// Form stays open.
109         /// The quantity equals 0
110         /// </summary>
111         private void btnMaterialsSell_Click(object sender, EventArgs e)
112         {
113             for (int i = 1; i <= TheInventory.DataInventoryHALF.Count; i++)
114             {
115                 if (((RadioButton)tlpMaterials.Controls["rdbMaterial" + (i)]).Checked)
116                 {
117                     if (TheInventory.DataInventoryHALF.ContainsKey(((RadioButton)
                        tlpMaterials.Controls["rdbMaterial" + (i)]).Text))
118                     {
119                         string myKey = ((RadioButton)tlpMaterials.Controls[
                            "rdbMaterial" + (i)]).Text;
120 
121                         TheInventory.DataInventoryFULL[myKey] = 0;
122                         TheInventory.DataInventoryHALF[myKey] = 0;
123                     }
124                     TheInventory.Rewrite();
125                 }
126 
127             }
128             UpdateView();
129         }
130 
131         private void btnMaterialsBack_Click(object sender, EventArgs e)
132         {
133             this.Close();
134         }
135     }
136 }
137 
```

```
/*************************************************************************
 * Author       : Jessica Sulzbach
 * Class        : I.In-P4B
 * Project      : TPI - Virtual animal
 * Name         : VirtualAnimalStore
 * Description  : Store form
 * Last modified: 23.05.2017
 *************************************************************************/
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace VirtualAnimal
{
    public partial class VirtualAnimalStore : Form
    {
        #region Variables
        // Variables
        private Store _theStore;
        private const int NUMBER_OF_PRODUCTS = 7;
        public int TotalPrice = 0;
        #endregion

        #region Properties
        // Properties
        public Store TheStore
        {
            get { return _theStore; }
            set { _theStore = value; }
        }
        #endregion

        public VirtualAnimalStore()
        {
            InitializeComponent();
            TheStore = new Store();
        }

        private void VirtualAnimalStore_Load(object sender, EventArgs e)
        {
            UpdateView();
        }

        #region Methods
        //Methods

        /// <summary>
        /// Fills the table panel layout with the labels and textboxes necessary
        /// </summary>
        private void UpdateView()
        {
```

```
58              tlpStore.Controls.Clear();
59
60              int Column = 1;
61
62              // Titles
63              tlpStore.Controls.Add(new Label() { Text = "Produits", Anchor =
                AnchorStyles.None, AutoSize = true, Font = new Font("Verdana", 12,
                FontStyle.Bold) }, 0, 0);
64              tlpStore.Controls.Add(new Label() { Text = "Prix", Anchor = AnchorStyles.
                None, AutoSize = true, Font = new Font("Verdana", 12, FontStyle.Bold) },
                1, 0);
65              tlpStore.Controls.Add(new Label() { Text = "Qnt", Anchor = AnchorStyles.
                None, AutoSize = true, Font = new Font("Verdana", 12, FontStyle.Bold) },
                2, 0);
66
67              // Products and price
68              foreach (var pair in TheStore.DataStore)
69              {
70                  tlpStore.Controls.Add(new Label() { Text = pair.Key, Anchor =
                    AnchorStyles.Left, AutoSize = true, Font = new Font("Verdana", 12,
                    FontStyle.Regular) }, 0, Column);
71                  tlpStore.Controls.Add(new Label() { Text = string.Format("{0:0.00}",
                    pair.Value), Anchor = AnchorStyles.None, AutoSize = true, Font = new
                    Font("Verdana", 12, FontStyle.Regular) }, 1, Column);
72                  tlpStore.Controls.Add(new TextBox() { Name = "tbxQnty" + Column }, 2,
                     Column);
73                  Column++;
74              }
75
76              // Adds evenements
77              for (int i = 0; i < NUMBER_OF_PRODUCTS; i++)
78              {
79                  ((TextBox)tlpStore.Controls["tbxQnty" + (i + 1)]).KeyPress += new
                    KeyPressEventHandler(Filter);
80                  ((TextBox)tlpStore.Controls["tbxQnty" + (i + 1)]).KeyUp += new
                    KeyEventHandler(CalculateTolatPrice);
81              }
82
83              // Money
84              TheStore.Money();
85              this.lblMoney.Text = Convert.ToString(TheStore.SaveOrRecover.
                DataToRecover_Animal["Money"]);
86          }
87
88          /// <summary>
89          /// Calculates the total price each time a key is pressed up
90          /// </summary>
91          public void CalculateTolatPrice(object sender, KeyEventArgs e)
92          {
93              List<int> QntyList = new List<int>();
94              int[] t = new int[NUMBER_OF_PRODUCTS];
95
96              for (int i = 0; i < NUMBER_OF_PRODUCTS; i++)
97              {
98                  if (tlpStore.Controls["tbxQnty" + (i + 1)].Text == "")
99                  {
100                     t[i] = 0;
```

```
101                 }
102                 else
103                 {
104                     t[i] = Convert.ToInt32(((TextBox)tlpStore.Controls["tbxQnty" + (i
                        + 1)]).Text);
105                 }
106 
107                 QntyList.Add(t[i]);
108             }
109 
110             int listIndex = 0;
111             TotalPrice = 0;
112             foreach (var pair in TheStore.DataStore)
113             {
114                 TotalPrice += Convert.ToInt32(pair.Value * QntyList[listIndex]);
115                 listIndex++;
116             }
117 
118             this.lblPrice.Text = Convert.ToString(TotalPrice);
119         }
120 
121         /// <summary>
122         /// Filters the input allowed in the textBox
123         /// </summary>
124         public void Filter(object sender, KeyPressEventArgs e)
125         {
126             if (char.IsLetter(e.KeyChar) || char.IsSymbol(e.KeyChar) || char.
                IsWhiteSpace(e.KeyChar) || char.IsPunctuation(e.KeyChar)) e.Handled =
                true;
127         }
128 
129         public void Buy()
130         {
131             List<int> QntyList = new List<int>();
132             int numberOfZero = 0;
133             int[] t = new int[NUMBER_OF_PRODUCTS];
134             for (int i = 0; i < NUMBER_OF_PRODUCTS; i++)
135             {
136                 if (tlpStore.Controls["tbxQnty" + (i + 1)].Text == "")
137                 {
138                     t[i] = 0;
139                     numberOfZero++;
140                 }
141                 else
142                 {
143                     t[i] = Convert.ToInt32(((TextBox)tlpStore.Controls["tbxQnty" + (i
                        + 1)]).Text);
144                 }
145 
146                 QntyList.Add(t[i]);
147                 ((TextBox)tlpStore.Controls["tbxQnty" + (i + 1)]).Text = "";
148 
149             }
150 
151             if (numberOfZero < NUMBER_OF_PRODUCTS && TheStore.SaveOrRecover.
                DataToRecover_Animal["Money"] - TotalPrice >= 0)
152             {
```

```
153                 this.TheStore.Sell(QntyList, TotalPrice);
154
155             }
156             else if (TheStore.SaveOrRecover.DataToRecover_Animal["Money"] -
                TotalPrice < 0)
157             {
158                 MessageBox.Show("Désolé, mais vous n'avez pas azzes d'argent...
                    Allez faire une promenade pour trouver plus d'argent!", "Attention",
                    MessageBoxButtons.OK, MessageBoxIcon.Information);
159             }
160         }
161
162         private void bntBack_Click(object sender, EventArgs e)
163         {
164             this.Close();
165         }
166
167         private void btnBuy_Click(object sender, EventArgs e)
168         {
169             this.Buy();
170         }
171
172         private void btnBuyAndBack_Click(object sender, EventArgs e)
173         {
174             this.Buy();
175             this.Close();
176         }
177         #endregion
178     }
179 }
180
```

```
/*************************************************************************
 * Author       : Jessica Sulzbach
 * Class        : I.In-P4B
 * Project      : TPI - Virtual animal
 * Name         : Animal
 * Description  : This class manages the life levels, money and animations
 * Last modified: 23.05.2017
 *************************************************************************/
using System;
using System.Collections.Generic;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace VirtualAnimal
{
    public class Animal
    {
        #region Variables
        // Variables
        // Niveaux de vie
        private int _hygene;
        private int _energy;
        private int _health;
        private int _happiness;

        // Monnaie
        private double _money;

        // Nom et age
        private string name;
        private int age;

        // Animation
        private List<Image> _anim;
        private int numImage;
        // Promenade
        private Dictionary<int, double> gifts = new Dictionary<int, double>();

        // Class de reucperation et enregistrement
        private DataRecovery _saveOrRecover;
        #endregion

        #region Properties
        // Properties
        public int Hygene
        {
            get { return _hygene; }
            set { _hygene = value; }
        }
        public int Energy
        {
            get { return _energy; }
            set { _energy = value; }
        }
        public int Happiness
```

```
58          {
59              get { return _happiness; }
60              set { _happiness = value; }
61          }
62          public double Money
63          {
64              get { return _money; }
65              set { _money = value; }
66          }
67          public int Health
68          {
69              get { return _health; }
70              set { _health = value; }
71          }
72          public DataRecovery SaveOrRecover
73          {
74              get { return _saveOrRecover; }
75              set { _saveOrRecover = value; }
76          }
77
78          public int NumImage
79          {
80              get { return numImage; }
81              set { numImage = value; }
82          }
83
84          public Dictionary<int, double> Gifts
85          {
86              get { return gifts; }
87              set { gifts = value; }
88          }
89
90          public int Age
91          {
92              get { return age; }
93              set { age = value; }
94          }
95
96          public string Name
97          {
98              get{ return name; }
99              set{ name = value; }
100         }
101         public List<Image> Anim
102         {
103             get { return _anim; }
104             set { _anim = value; }
105         }
106         #endregion
107
108         #region Constructor
109         // Constructor
110         public Animal()
111         {
112             SaveOrRecover = new DataRecovery();
113             Anim = new List<Image>();
114             AnimalRecover();
```

```
115         }
116         #endregion
117
118         #region Methods
119         // Methods
120
121         /// <summary>
122         /// Recovers the data and initiates the properties
123         /// </summary>
124         public void AnimalRecover()
125         {
126             SaveOrRecover.FileReader("Save_Animal.txt");
127             this.Health = SaveOrRecover.DataToRecover_Animal["Health"];
128             this.Hygene = SaveOrRecover.DataToRecover_Animal["Hygene"];
129             this.Energy = SaveOrRecover.DataToRecover_Animal["Energy"];
130             this.Happiness = SaveOrRecover.DataToRecover_Animal["Happiness"];
131             this.Money = SaveOrRecover.DataToRecover_Animal["Money"];
132
133             SaveOrRecover.FileReader("Animal_Age_Name.txt");
134             this.Age=Convert.ToInt32(SaveOrRecover.SeperateData[1]);
135             this.Name = SaveOrRecover.SeperateData[0];
136         }
137
138         /// <summary>
139         /// Saves the data in the Save_Animal.txt
140         /// </summary>
141         /// <param name="Save">Text to be writen in text file</param>
142         public void AnimalSave(int Save)
143         {
144             SaveOrRecover.FileWritter("Save_Animal.txt", Convert.ToString(Save));
145         }
146
147         /// <summary>
148         /// According to the animation name, the images of the animation will be
            added to the image list
149         /// </summary>
150         /// <param name="Animation">Animation name</param>
151         public void Animations(string Animation)
152         {
153             switch (Animation)
154             {
155                 case "Idle":
156                     {
157                         NumImage = 0;
158                         this.Anim.Clear();
159                         for (int i = 0; i <= 2; i++)
160                         {
161                             string ImageName = "Idle" + i;
162                             this.Anim.Add((Image)Properties.Resources.ResourceManager
                                .GetObject(ImageName, Properties.Resources.Culture));
163                         }
164                         break;
165                     }
166                 case "Walk":
167                     {
168                         Gifts.Clear();
169                         Random rSec = new Random();
```

```
170                         Random r = new Random();
171                         int numberOfGifts = r.Next(1, 6);
172
173                         for (int i = 0; i < numberOfGifts; i++)
174                         {
175                             int key = rSec.Next(1, 26);
176                             if (!Gifts.ContainsKey(key))
177                             {
178                                 Gifts.Add(key, r.Next(5, 51));
179                             }
180                             else
181                             {
182                                 i--;
183                             }
184                         }
185
186                         NumImage = 0;
187                         this.Anim.Clear();
188                         this.Anim.Add(Properties.Resources.walk0);
189                         this.Anim.Add(Properties.Resources.walk1);
190
191                         if (this.Happiness + 30 > 100)
192                         {
193                             this.Happiness = 100;
194                         }
195                         else
196                         {
197                             this.Happiness = this.Happiness + 30;
198                         }
199                         if (this.Hygene - 30 < 0)
200                         {
201                             this.Hygene = 0;
202                         }
203                         else
204                         {
205                             this.Hygene = this.Hygene - 30;
206                         }
207                         if (this.Energy - 30 < 0)
208                         {
209                             this.Energy = 0;
210                         }
211                         else
212                         {
213                             this.Energy = this.Energy - 30;
214                         }
215                         break;
216                     }
217                 case "Happy":
218                     {
219                         NumImage = 0;
220                         this.Anim.Clear();
221                         for (int i = 0; i <= 8; i++)
222                         {
223                             string ImageName = "Happy" + i;
224                             this.Anim.Add((Image)Properties.Resources.ResourceManager
                                 .GetObject(ImageName, Properties.Resources.Culture));
225                         }
```

```
226                         if (this.Happiness + 20 > 100)
227                         {
228                             this.Happiness = 100;
229                         }
230                         else
231                         {
232                             this.Happiness = this.Happiness + 20;
233                         }
234                         break;
235                     }
236                 case "Eat":
237                     {
238                         NumImage = 0;
239                         this.Anim.Clear();
240                         this.Anim.Add(Properties.Resources.Eat1);
241                         this.Anim.Add(Properties.Resources.Eat2);
242                         if (this.Health + 40 > 100)
243                         {
244                             this.Health = 100;
245                         }
246                         else
247                         {
248                             this.Health = this.Health + 40;
249                         }
250                         break;
251                     }
252                 case "Dead":
253                     {
254                         break;
255                     }
256                 case "Shower":
257                     {
258                         NumImage = 0;
259                         this.Anim.Clear();
260                         for (int i = 0; i <= 7; i++)
261                         {
262                             string ImageName = "Shower" + i;
263                             this.Anim.Add((Image)Properties.Resources.ResourceManager
                                .GetObject(ImageName, Properties.Resources.Culture));
264                         }
265                         if (this.Hygene + 40 > 100)
266                         {
267                             this.Hygene = 100;
268                         }
269                         else
270                         {
271                             this.Hygene = this.Hygene + 40;
272                         }
273                         break;
274                     }
275                 case "Brush":
276                     {
277                         NumImage = 0;
278                         Anim.Clear();
279                         this.Anim.Add(Properties.Resources.brush0);
280                         this.Anim.Add(Properties.Resources.brush1);
281
```

```
282                         if (this.Hygene + 20 > 100)
283                         {
284                             this.Hygene = 100;
285                         }
286                         else
287                         {
288                             this.Hygene = this.Hygene + 20;
289                         }
290                         break;
291                     }
292                 case "Sleep":
293                     {
294                         NumImage = 0;
295                         this.Anim.Clear();
296                         for (int i = 0; i <= 18; i++)
297                         {
298                             string ImageName = "Sleep_Nap" + i;
299                             this.Anim.Add((Image)Properties.Resources.ResourceManager
                                .GetObject(ImageName, Properties.Resources.Culture));
300                         }
301                         this.Energy = 100;
302                         break;
303                     }
304                 case "Nap":
305                     {
306                         NumImage = 0;
307                         this.Anim.Clear();
308                         for (int i = 0; i <= 18; i++)
309                         {
310                             string ImageName = "Sleep_Nap" + i;
311                             this.Anim.Add((Image)Properties.Resources.ResourceManager
                                .GetObject(ImageName, Properties.Resources.Culture));
312                         }
313                         if (this.Energy + 40 > 100)
314                         {
315                             this.Energy = 100;
316                         }
317                         else
318                         {
319                             this.Energy = this.Energy + 40;
320                         }
321                         break;
322                     }
323                 case "Death":
324                     {
325                         this.Anim.Clear();
326                         for (int i = 0; i <= 4; i++)
327                         {
328                             string ImageName = "death" + i;
329                             this.Anim.Add((Image)Properties.Resources.ResourceManager
                                .GetObject(ImageName, Properties.Resources.Culture));
330                         }
331                         break;
332                     }
333                 case "Born":
334                     {
335                         this.Anim.Clear();
```

```
336                         for (int i = 0; i <= 8; i++)
337                         {
338                             string ImageName = "Born" + i;
339                             this.Anim.Add((Image)Properties.Resources.ResourceManager
                                .GetObject(ImageName, Properties.Resources.Culture));
340                         }
341                         this.Energy = 100;
342                         this.Happiness = 100;
343                         this.Health = 100;
344                         this.Hygene = 100;
345                         this.Money = 100;
346 
347                         break;
348                     }
349                 }
350             }
351             #endregion
352         }
353     }
354 
```

```
1   /*************************************************************************
2    * Author       : Jessica Sulzbach
3    * Class        : I.In-P4B
4    * Project      : TPI - Virtual animal
5    * Name         : Inventory
6    * Description  : This class manages the add and subtract of products
7    * Last modified: 23.05.2017
8    *************************************************************************/
9   using System;
10  using System.Collections.Generic;
11  using System.Linq;
12  using System.Text;
13  using System.Threading.Tasks;
14
15  namespace VirtualAnimal
16  {
17      public class Inventory
18      {
19          #region Variables
20          // Variables
21          private DataRecovery _saveOrRecover;
22          private Dictionary<string, int> _dataInventoryHALF;
23          private Dictionary<string, int> _dataInventoryFULL;
24          private string _product;
25          #endregion
26
27          #region Properties
28          // Properties
29          public string Product
30          {
31              get { return _product; }
32              set { _product = value; }
33          }
34
35          public const int LAST_OF_FOODS = 8;
36
37          public Dictionary<string, int> DataInventoryFULL
38          {
39              get { return _dataInventoryFULL; }
40              set { _dataInventoryFULL = value; }
41          }
42          public const int LAST_OF_MATERIALS = 20;
43
44          public Dictionary<string, int> DataInventoryHALF
45          {
46              get { return _dataInventoryHALF; }
47              set { _dataInventoryHALF = value; }
48          }
49
50          public DataRecovery SaveOrRecover
51          {
52              get { return _saveOrRecover; }
53              set { _saveOrRecover = value; }
54          }
55          #endregion
56
57          #region Constructor
```

```
58          // Constructor
59          public Inventory()
60          {
61              // Initiates
62              SaveOrRecover = new DataRecovery();
63              DataInventoryHALF = new Dictionary<string, int>();
64              DataInventoryFULL = new Dictionary<string, int>();
65
66              SaveOrRecover.FileReader("Product_name_and_price.txt");
67          }
68          #endregion
69
70          #region Methods
71          // Methods
72          /// <summary>
73          /// Adds all the products data to the list
74          /// </summary>
75          public void InventoryData()
76          {
77              DataInventoryFULL.Clear();
78              for (int i = 0; i < SaveOrRecover.SeperateData.Count - 3; i = i + 3)
79              {
80                  for (int y = 2; y <= SaveOrRecover.SeperateData.Count - 1; y = y + 3)
81                  {
82                      DataInventoryFULL.Add(SaveOrRecover.SeperateData[i], Convert.
                        ToInt32(SaveOrRecover.SeperateData[y]));
83                      if (i < SaveOrRecover.SeperateData.Count - 3)
84                      {
85                          i = i + 3;
86                      }
87                  }
88              }
89          }
90
91          /// <summary>
92          /// Adds certain product data to the list
93          /// </summary>
94          /// <param name="NameView">Type of products</param>
95          public void InventoryData(string NameView)
96          {
97              DataInventoryHALF.Clear();
98              if (NameView == "Materials")
99              {
100                 for (int i = 9; i < LAST_OF_MATERIALS - 2; i = i + 3)
101                 {
102                     for (int y = 11; y <= LAST_OF_MATERIALS; y = y + 3)
103                     {
104                         DataInventoryHALF.Add(SaveOrRecover.SeperateData[i], Convert.
                            ToInt32(SaveOrRecover.SeperateData[y]));
105                         if (i < SaveOrRecover.SeperateData.Count - 3)
106                         {
107                             i = i + 3;
108                         }
109                     }
110                 }
111             }
112             if (NameView == "Food")
```

```
113             {
114                 for (int i = 0; i < LAST_OF_FOODS - 2; i = i + 3)
115                 {
116                     for (int y = 2; y <= LAST_OF_FOODS; y = y + 3)
117                     {
118                         DataInventoryHALF.Add(SaveOrRecover.SeperateData[i], Convert.
                            ToInt32(SaveOrRecover.SeperateData[y]));
119                         if (i < SaveOrRecover.SeperateData.Count - 3)
120                         {
121                             i = i + 3;
122                         }
123                     }
124                 }
125             }
126         }
127 
128         /// <summary>
129         /// Initiates the Product variable with the name of the products
130         /// This will be usefull when calling an animation
131         /// </summary>
132         /// <param name="ProductName">The selected products name</param>
133         public void Use(string ProductName)
134         {
135             Product = ProductName;
136         }
137 
138         /// <summary>
139         /// Rewrites the Save_Aniaml text file with the new quantity
140         /// </summary>
141         public void Rewrite()
142          {
143             SaveOrRecover.FirstTime = true;
144 
145             for (int i = 0; i <= SaveOrRecover.SeperateData.Count - 3; i++)
146             {
147                 for (int y = 1; y <= SaveOrRecover.SeperateData.Count - 2; y++)
148                 {
149                     foreach (var pair in DataInventoryFULL)
150                     {
151                         string FinalLineToSave = string.Format("{0};{1};{2}",
                            SaveOrRecover.SeperateData[i], SaveOrRecover.SeperateData[y],
                             pair.Value);
152                         SaveOrRecover.FileWritter("Product_name_and_price.txt",
                            FinalLineToSave);
153                         if (i < SaveOrRecover.SeperateData.Count - 3)
154                         {
155                             i = i + 3;
156                             y = y + 3;
157                         }
158                     }
159                 }
160             }
161         }
162 
163         /// <summary>
164         /// Rewrites the Save_Aniaml text file with all quantities at 0
165         /// For starting new game
```

```
166         /// </summary>
167         public void RewriteNew()
168         {
169             InventoryData();
170             SaveOrRecover.FirstTime = true;
171 
172             for (int i = 0; i <= SaveOrRecover.SeperateData.Count - 3; i++)
173             {
174                 for (int y = 1; y <= SaveOrRecover.SeperateData.Count - 2; y++)
175                 {
176                     foreach (var pair in DataInventoryFULL)
177                     {
178                         string FinalLineToSave = string.Format("{0};{1};{2}",
                            SaveOrRecover.SeperateData[i], SaveOrRecover.SeperateData[y],
                             0);
179                         SaveOrRecover.FileWritter("Product_name_and_price.txt",
                            FinalLineToSave);
180                         if (i < SaveOrRecover.SeperateData.Count - 3)
181                         {
182                             i = i + 3;
183                             y = y + 3;
184                         }
185                     }
186                 }
187             }
188         }
189         #endregion
190     }
191 }
192 
```

```
/*****************************************************************************
 * Author       : Jessica Sulzbach
 * Class        : I.In-P4B
 * Project      : TPI - Virtual animal
 * Name         : Store
 * Description  : This class wroks with the Inventory and manages
 *                the products being bought from the store and the money
 * Last modified: 23.05.2017
 *****************************************************************************/
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace VirtualAnimal
{
    public class Store
    {
        #region Variables
        // Variables
        private DataRecovery _saveOrRecover;
        private Dictionary<string, double> _dataStore;
        #endregion

        #region Properties
        // Properties
        public Dictionary<string, double> DataStore
        {
            get { return _dataStore; }
            set { _dataStore = value; }
        }

        public DataRecovery SaveOrRecover
        {
            get { return _saveOrRecover; }
            set { _saveOrRecover = value; }
        }
        #endregion

        #region Constructor
        // Constructor
        public Store()
        {
            // Initiates
            SaveOrRecover = new DataRecovery();
            DataStore = new Dictionary<string, double>();

            // Read the Product_name_and_price test file for the data
            SaveOrRecover.FileReader("Product_name_and_price.txt");

            StoreData();
        }
        #endregion

        #region Methods
        // Methods
```

```
58
59          /// <summary>
60          /// Prepares the the data we nedd for the store
61          /// Only the products name and price
62          /// </summary>
63          public void StoreData()
64          {
65              DataStore.Clear();
66              for (int i = 0; i < SaveOrRecover.SeperateData.Count - 2; i = i + 3)
67              {
68                  for (int y = 1; y < SaveOrRecover.SeperateData.Count; y = y + 3)
69                  {
70                      DataStore.Add(SaveOrRecover.SeperateData[i], Convert.ToDouble(
                        SaveOrRecover.SeperateData[y]));
71                      if (i < SaveOrRecover.SeperateData.Count - 3)
72                      {
73                          i = i + 3;
74                      }
75                  }
76              }
77          }
78
79          /// <summary>
80          /// Adds new and old quantity
81          /// New quantity is saved
82          /// Subtracts the total price from the money
83          /// New money amount saved
84          /// </summary>
85          /// <param name="NewQuantity">List of the new quantitites</param>
86          /// <param name="TotalCost">Price of groceries</param>
87          public void Sell(List<int> NewQuantity, int TotalCost)
88          {
89              SaveOrRecover.FileReader("Product_name_and_price.txt");
90              SaveOrRecover.FirstTime = true;
91
92              List<int> OldQuantity = new List<int>();
93              for (int i = 2; i < SaveOrRecover.SeperateData.Count; i = i + 3)
94              {
95                  OldQuantity.Add(Convert.ToInt32(SaveOrRecover.SeperateData[i]));
96              }
97              // New quantity
98              for (int i = 0; i < NewQuantity.Count; i++)
99              {
100                 NewQuantity[i] = NewQuantity[i] + OldQuantity[i];
101             }
102             // Saveing quantity
103             for (int i = 0; i < SaveOrRecover.SeperateData.Count - 2; i++)
104             {
105                 for (int y = 1; y < SaveOrRecover.SeperateData.Count - 1; y++)
106                 {
107                     for (int j = 0; j <= NewQuantity.Count - 1; j++)
108                     {
109                         string FinalLineToSave = string.Format("{0};{1};{2}",
                            SaveOrRecover.SeperateData[i], SaveOrRecover.SeperateData[y],
                             NewQuantity[j]);
110                         SaveOrRecover.FileWritter("Product_name_and_price.txt",
                            FinalLineToSave);
```

```
111                         if (i < SaveOrRecover.SeperateData.Count - 3)
112                         {
113                             i = i + 3;
114                             y = y + 3;
115                         }
116                     }
117                 }
118             }
119 
120             // Money
121             SaveOrRecover.FirstTime = true;
122             SaveOrRecover.DataToRecover_Animal["Money"] = SaveOrRecover.
                DataToRecover_Animal["Money"] - TotalCost;
123             foreach (var pair in SaveOrRecover.DataToRecover_Animal)
124             {
125                 SaveOrRecover.FileWritter("Save_Animal.txt", Convert.ToString(pair.
                    Value));
126             }
127 
128 
129         }
130 
131         /// <summary>
132         /// Reads the Save_animal text file for the data
133         /// </summary>
134         public void Money()
135         {
136             SaveOrRecover.FileReader("Save_Animal.txt");
137         }
138         #endregion
139     }
140 }
141 
```

```
/*****************************************************************************
 * Author       : Jessica Sulzbach
 * Class        : I.In-P4B
 * Project      : TPI - Virtual animal
 * Name         : DataRecovery
 * Description  : This class recuperates the data and rewrite it
 * Last modified: 23.05.2017
 *****************************************************************************/
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
using System.Threading.Tasks;

namespace VirtualAnimal
{
    public class DataRecovery
    {
        #region Variables
        // Variables
        // Bool to append or not
        private bool _firstTime;
        // List where the seperate ( without ; ) data is added
        private List<string> _seperateData;
        private Dictionary<string, int> _dataToRecover_Animal;

        #endregion

        #region Properties
        // Properties
        public Dictionary<string, int> DataToRecover_Animal
        {
            get { return _dataToRecover_Animal; }
            set { _dataToRecover_Animal = value; }
        }
        public bool FirstTime
        {
            get { return _firstTime; }
            set { _firstTime = value; }
        }
        public List<string> SeperateData
        {
            get { return _seperateData; }
            set { _seperateData = value; }
        }
        #endregion

        #region Constructor
        // Constructor
        public DataRecovery()
        {
            FirstTime = true;
            //Initiate
            DataToRecover_Animal = new Dictionary<string, int>();
            SeperateData = new List<string>();
        }
```

```
58          #endregion
59
60          /// <summary>
61          /// Writtes in the text files
62          /// </summary>
63          /// <param name="FileName">The files name</param>
64          /// <param name="DataToSave">Test to be writen</param>
65          public void FileWritter(string FileName, string DataToSave)
66          {
67              StreamWriter Write;
68              // The text file will be cleared before writing
69              if (FirstTime == true)
70              {
71                  Write = new StreamWriter(@"..\\..\\Resources\" + FileName, false);
72                  Write.WriteLine(DataToSave);
73                  Write.Close();
74                  FirstTime = false;
75              }
76              // The text file will not be cleared before writing
77              // The text will be added at the end of the text file
78              else
79              {
80                  Write = new StreamWriter(@"..\\..\\Resources\" + FileName, true);
81                  Write.WriteLine(DataToSave);
82                  Write.Close();
83              }
84          }
85
86          /// <summary>
87          /// Reads texte files and according to the file name saves the data in lists
88          /// and then closes the file
89          /// </summary>
90          /// <param name="FileName">The files name</param>
91          public void FileReader(string FileName)
92          {
93              StreamReader Read;
94              Read = new StreamReader(@"..\\..\\Resources\" + FileName);
95              // List of the data by line
96              List<string> Data = new List<string>();
97              SeperateData.Clear();
98              string LineBeingRead;
99
100             if (FileName == "Save_Animal.txt")
101             {
102                 Data.Clear();
103
104                 while ((LineBeingRead = Read.ReadLine()) != null)
105                 {
106                     Data.Add(LineBeingRead);
107                 }
108
109                 Read.Close();
110
111                 DataToRecover_Animal.Add("Health", Convert.ToInt32(Data[0]));
112                 DataToRecover_Animal.Add("Hygene", Convert.ToInt32(Data[1]));
113                 DataToRecover_Animal.Add("Energy", Convert.ToInt32(Data[2]));
114                 DataToRecover_Animal.Add("Happiness", Convert.ToInt32(Data[3]));
```

```
115                 DataToRecover_Animal.Add("Money", Convert.ToInt32(Data[4]));
116             }
117             if (FileName == "Product_name_and_price.txt")
118             {
119                 while ((LineBeingRead = Read.ReadLine()) != null)
120                 {
121                     Data.Add(LineBeingRead);
122                 }
123 
124                 foreach (var item in Data)
125                 {
126                     SeperateData.AddRange(item.Split(';').ToList());
127                 }
128 
129                 Read.Close();
130             }
131             if (FileName == "Animal_Age_Name.txt")
132             {
133                 while ((LineBeingRead = Read.ReadLine()) != null)
134                 {
135                     SeperateData.Add(LineBeingRead);
136                 }
137 
138                 Read.Close();
139             }
140         }
141     }
142 }
143 
```