Eyes of the Dragon Tutorials
Part 62
Back to the Future, uhm, Editors. I Meant Editors.

I'm writing these tutorials for the MonoGame 3.8 framework using Visual Studio 2019. Though you cannot create MonoGame projects with Visual Studio 2022, the project can be opened and run. So, if you have downloaded the project from GitHub, or you've been following along, you can use Visual Studio 2022. The tutorials will make more sense if they are read in order. You can find the list of tutorials on the Eyes of the Dragon page of my web blog. I will be making each version of the project available on GitHub here. It will be included on the page that links to the tutorials.

Okay, corny joke aside, in this tutorial I will be diving back into the editors. For a role playing game, you need lots of data. While you could hard code it, it is better to be able to generate the content and modify it easily. For that reason, I will be extending to the editor to create spells and talents. Boot up your project from last time.

I started with a form for spells and spell details. I didn't add a form for creating the effects for a spell. I will start with that. First, right click the RpgEditor project and select Set As Start Up. Right click the MGRpgLibrary project, select Add and Form (Windows Forms). Name this new form FormBaseEffect. Now, expand the node beside the BaseEffectForm and replace the FormBaseEffect.Designer.cs file with the following code.

```
namespace RpgEditor
{
    partial class FormBaseEffect
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
```

```
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.label1 = new System.Windows.Forms.Label();
    this.tbName = new System.Windows.Forms.TextBox();
    this.groupBox1 = new System.Windows.Forms.GroupBox();
    this.rbHealing = new System.Windows.Forms.RadioButton();
    this.rbDamage = new System.Windows.Forms.RadioButton();
    this.label2 = new System.Windows.Forms.Label();
    this.cboDamage = new System.Windows.Forms.ComboBox();
    this.label3 = new System.Windows.Forms.Label();
    this.cboTarget = new System.Windows.Forms.ComboBox();
    this.cboDieType = new System.Windows.Forms.ComboBox();
    this.label4 = new System.Windows.Forms.Label();
    this.label5 = new System.Windows.Forms.Label();
    this.sbNumOfDice = new System.Windows.Forms.NumericUpDown();
    this.btnOK = new System.Windows.Forms.Button();
    this.btnCancel = new System.Windows.Forms.Button();
    this.label6 = new System.Windows.Forms.Label();
    this.sbModifier = new System.Windows.Forms.NumericUpDown();
    this.groupBox1.SuspendLayout();
    ((System.ComponentModel.ISupportInitialize)(this.sbNumOfDice)).BeginInit();
    ((System.ComponentModel.ISupportInitialize)(this.sbModifier)).BeginInit();
    this.SuspendLayout();
    //
    // label1
    //
    this.label1.AutoSize = true;
    this.label1.Location = new System.Drawing.Point(12, 9);
    this.label1.Name = "label1";
    this.label1.Size = new System.Drawing.Size(38, 13);
    this.label1.TabIndex = 0;
    this.label1.Text = "Name:";
    //
    // tbName
    //
    this.tbName.Location = new System.Drawing.Point(56, 6);
    this.tbName.Name = "tbName";
    this.tbName.Size = new System.Drawing.Size(110, 20);
    this.tbName.TabIndex = 1;
    //
    // groupBox1
    //
    this.groupBox1.Controls.Add(this.rbHealing);
    this.groupBox1.Controls.Add(this.rbDamage);
    this.groupBox1.Location = new System.Drawing.Point(12, 32);
    this.groupBox1.Name = "groupBox1";
    this.groupBox1.Size = new System.Drawing.Size(84, 73);
    this.groupBox1.TabIndex = 2;
    this.groupBox1.TabStop = false;
    this.groupBox1.Text = "Effect Type";
    //
```

```
// rbHealing
//
this.rbHealing.AutoSize = true;
this.rbHealing.Location = new System.Drawing.Point(6, 42);
this.rbHealing.Name = "rbHealing";
this.rbHealing.Size = new System.Drawing.Size(61, 17);
this.rbHealing.TabIndex = 1;
this.rbHealing.Text = "Healing";
this.rbHealing.UseVisualStyleBackColor = true;
//
// rbDamage
//
this.rbDamage.AutoSize = true;
this.rbDamage.Checked = true;
this.rbDamage.Location = new System.Drawing.Point(6, 19);
this.rbDamage.Name = "rbDamage";
this.rbDamage.Size = new System.Drawing.Size(65, 17);
this.rbDamage.TabIndex = 0;
this.rbDamage.TabStop = true;
this.rbDamage.Text = "Damage";
this.rbDamage.UseVisualStyleBackColor = true;
//
// label2
//
this.label2.AutoSize = true;
this.label2.Location = new System.Drawing.Point(9, 111);
this.label2.Name = "label2";
this.label2.Size = new System.Drawing.Size(50, 13);
this.label2.TabIndex = 3;
this.label2.Text = "Damage:";
//
// cboDamage
//
this.cboDamage.FormattingEnabled = true;
this.cboDamage.Location = new System.Drawing.Point(72, 108);
this.cboDamage.Name = "cboDamage";
this.cboDamage.Size = new System.Drawing.Size(94, 21);
this.cboDamage.TabIndex = 4;
//
// label3
//
this.label3.AutoSize = true;
this.label3.Location = new System.Drawing.Point(9, 138);
this.label3.Name = "label3";
this.label3.Size = new System.Drawing.Size(41, 13);
this.label3.TabIndex = 5;
this.label3.Text = "Target:";
//
// cboTarget
//
this.cboTarget.FormattingEnabled = true;
this.cboTarget.Location = new System.Drawing.Point(72, 135);
this.cboTarget.Name = "cboTarget";
this.cboTarget.Size = new System.Drawing.Size(94, 21);
```

```
this.cboTarget.TabIndex = 6;
//
// cboDieType
//
this.cboDieType.FormattingEnabled = true;
this.cboDieType.Location = new System.Drawing.Point(72, 162);
this.cboDieType.Name = "cboDieType";
this.cboDieType.Size = new System.Drawing.Size(94, 21);
this.cboDieType.TabIndex = 7;
//
// label4
//
this.label4.AutoSize = true;
this.label4.Location = new System.Drawing.Point(9, 165);
this.label4.Name = "label4";
this.label4.Size = new System.Drawing.Size(53, 13);
this.label4.TabIndex = 8;
this.label4.Text = "Die Type:";
//
// label5
//
this.label5.AutoSize = true;
this.label5.Location = new System.Drawing.Point(12, 191);
this.label5.Name = "label5";
this.label5.Size = new System.Drawing.Size(54, 13);
this.label5.TabIndex = 9;
this.label5.Text = "# of Dice:";
//
// sbNumOfDice
//
this.sbNumOfDice.Location = new System.Drawing.Point(72, 189);
this.sbNumOfDice.Minimum = new decimal(new int[] {
1,
0,
0,
0});
this.sbNumOfDice.Name = "sbNumOfDice";
this.sbNumOfDice.Size = new System.Drawing.Size(94, 20);
this.sbNumOfDice.TabIndex = 10;
this.sbNumOfDice.Value = new decimal(new int[] {
1,
0,
0,
0});
//
// btnOK
//
this.btnOK.Location = new System.Drawing.Point(172, 4);
this.btnOK.Name = "btnOK";
this.btnOK.Size = new System.Drawing.Size(75, 23);
this.btnOK.TabIndex = 11;
this.btnOK.Text = "OK";
this.btnOK.UseVisualStyleBackColor = true;
//
```

```
// btnCancel
//
this.btnCancel.Location = new System.Drawing.Point(172, 32);
this.btnCancel.Name = "btnCancel";
this.btnCancel.Size = new System.Drawing.Size(75, 23);
this.btnCancel.TabIndex = 11;
this.btnCancel.Text = "Cancel";
this.btnCancel.UseVisualStyleBackColor = true;
//
// label6
//
this.label6.AutoSize = true;
this.label6.Location = new System.Drawing.Point(12, 217);
this.label6.Name = "label6";
this.label6.Size = new System.Drawing.Size(13, 13);
this.label6.TabIndex = 9;
this.label6.Text = "a";
//
// sbModifier
//
this.sbModifier.Location = new System.Drawing.Point(72, 215);
this.sbModifier.Name = "sbModifier";
this.sbModifier.Size = new System.Drawing.Size(94, 20);
this.sbModifier.TabIndex = 10;
//
// FormBaseEffect
//
this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
this.ClientSize = new System.Drawing.Size(257, 240);
this.Controls.Add(this.btnCancel);
this.Controls.Add(this.btnOK);
this.Controls.Add(this.sbModifier);
this.Controls.Add(this.label6);
this.Controls.Add(this.sbNumOfDice);
this.Controls.Add(this.label5);
this.Controls.Add(this.label4);
this.Controls.Add(this.cboDieType);
this.Controls.Add(this.cboTarget);
this.Controls.Add(this.label3);
this.Controls.Add(this.cboDamage);
this.Controls.Add(this.label2);
this.Controls.Add(this.groupBox1);
this.Controls.Add(this.tbName);
this.Controls.Add(this.label1);
this.MaximizeBox = false;
this.MinimizeBox = false;
this.Name = "FormBaseEffect";
this.Text = "FormBaseEffect";
this.groupBox1.ResumeLayout(false);
this.groupBox1.PerformLayout();
((System.ComponentModel.ISupportInitialize)(this.sbNumOfDice)).EndInit();
((System.ComponentModel.ISupportInitialize)(this.sbModifier)).EndInit();
this.ResumeLayout(false);
```

```
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.TextBox tbName;
        private System.Windows.Forms.GroupBox groupBox1;
        private System.Windows.Forms.RadioButton rbHealing;
        private System.Windows.Forms.RadioButton rbDamage;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.ComboBox cboDamage;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.ComboBox cboTarget;
        private System.Windows.Forms.ComboBox cboDieType;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.Label label5;
        private System.Windows.Forms.NumericUpDown sbNumOfDice;
        private System.Windows.Forms.Button btnOK;
        private System.Windows.Forms.Button btnCancel;
        private System.Windows.Forms.Label label6;
        private System.Windows.Forms.NumericUpDown sbModifier;
    }
}
```

This is all system generated code so I am not going to go over it. If you're interested, I'd suggest browsing it, but it is not necessary. Now, I will program the logic of the form. Save the form and then open the code view for the form. You can do that by selecting the form in the Solution Explorer and pressing the F7 key. Replace that with the following code.

```
using RpgLibrary;
using RpgLibrary.EffectClasses;
using RpgLibrary.SpellClasses;
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 RpgEditor
{
    public partial class FormBaseEffect : Form
    {
        public BaseEffectData BaseEffectData { get; set; }

        public FormBaseEffect()
        {
            InitializeComponent();
```

```
            btnOK.Click += BtnOK_Click;
            btnCancel.Click += BtnCancel_Click;

            Load += FormBaseEffect_Load;
            rbDamage.CheckedChanged += RbDamage_CheckedChanged;
        }

        private void BtnOK_Click(object sender, EventArgs e)
        {
            if (string.IsNullOrEmpty(tbName.Text))
            {
                MessageBox.Show("You must enter the name for the effect.");
                return;
            }

            if (rbDamage.Checked)
            {
                DamageEffectData data = new DamageEffectData();

                data.Name = tbName.Text;
                data.TargetType = (TargetType)Enum.Parse(typeof(TargetType),
cboTarget.SelectedItem.ToString());
                data.DamageType = (DamageType)Enum.Parse(typeof(DamageType),
cboDamage.SelectedItem.ToString());
                data.DieType = (DieType)Enum.Parse(typeof(DieType),
cboDieType.SelectedIndex.ToString());
                data.NumberOfDice = (int)sbNumOfDice.Value;
                data.Modifier = (int)sbModifier.Value;

                BaseEffectData = data;
            }
            else
            {
                HealEffectData data = new HealEffectData();

                data.Name = tbName.Text;
                data.TargetType = (TargetType)Enum.Parse(typeof(TargetType),
cboTarget.SelectedItem.ToString());
                data.DieType = (DieType)Enum.Parse(typeof(DieType),
cboDieType.SelectedIndex.ToString());
                data.NumberOfDice = (int)sbNumOfDice.Value;
                data.Modifier = (int)sbModifier.Value;

                BaseEffectData = data;
            }

            Close();
        }

        private void BtnCancel_Click(object sender, EventArgs e)
        {
            BaseEffectData = null;
            Close();
```

```
        }

        private void RbDamage_CheckedChanged(object sender, EventArgs e)
        {
            cboDamage.Enabled = rbDamage.Checked;
        }

        private void FormBaseEffect_Load(object sender, EventArgs e)
        {
            foreach (var v in Enum.GetNames(typeof(DamageType)))
            {
                cboDamage.Items.Add(v);
            }

            foreach (var v in Enum.GetNames(typeof(TargetType)))
            {
                cboTarget.Items.Add(v);
            }

            foreach (var v in Enum.GetValues(typeof(DieType)))
            {
                cboDieType.Items.Add(v);
            }
        }
    }
}
```

I added a property to the form that will hold the effect that was created. It is the base class so it can act as either of the derived types. In the constructor I wire handlers for the click events of the two buttons. Also, I wire handlers for the check changed of the radio button and the load event of the form.

In the click event handler for the OK button I check to make sure that a name has been entered into the text box. If it is empty, I display an error message and return from the method. If the damage radio button is the selected button I create a DamageEffectData instance and set the values to the values of the controls on the form. The only interesting thing is where I cast the enums using the Enum.Parse method. Otherwise, it is pretty straight forward. After creating the object I set the property to the object created. I do something similar for a heal effect otherwise. In either case, I close the form.

In the click event handler for the Cancel button all I do is set the property to null and close the form. In the check changed event handler for the damage radio button I set the enabled property of the damage type combo box to the Enabled property of the radio button as a visual cue that it is disabled for healing effects. In theory, you could have a heal property to remove a damage type such as if the player character was poisoned. That is an exercise that I will leave for you.

The last thing is the event handler for the load event of the form. It is here that I populate the combo boxes with their values. I do that using Enum.GetNames to get the text for each of the entries in the enumeration. Those names are then added to the appropriate combo box.

That is it for this form. Before I get to adding effects to spells I need to change the SpellData class.

Currently, it has a List<BaseEffect> for the effects of the spell. I want to change that. What I need to use is a List<BaseEffectData>. This is so that the spells can be serialized using the intermediate serializer. Replace the SpellData class with the following.

```
using RpgLibrary.EffectClasses;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace RpgLibrary.SpellClasses
{
    public enum SpellType { Passive, Sustained, Activated }
    public enum TargetType { Self, Enemy, Special }

    public class SpellData
    {
        #region Field Region

        public string Name;
        public string[] AllowedClasses;
        public Dictionary<string, int> AttributeRequirements;
        public string[] SpellPrerequisites;
        public int LevelRequirement;
        public SpellType SpellType;
        public int ActivationCost;
        public double CoolDown;
        public List<BaseEffectData> Effects;

        #endregion

        #region Property Region

        public double CastTime { get; set; }
        public double Duration { get; set; }
        public double Range { get; set; }
        public double AreaOfEffect { get; set; }
        public double AngleOfEffect { get; set; }

        #endregion

        #region Constructor Region

        public SpellData()
        {
            AttributeRequirements = new Dictionary<string, int>();
            Effects = new List<BaseEffectData>();
        }

        #endregion

        #region Method Region
        #endregion
```

```
        #region Virtual Method region

        public override string ToString()
        {
            string toString = Name;

            foreach (string s in AllowedClasses)
                toString += ", " + s;

            foreach (string s in AttributeRequirements.Keys)
                toString += ", " + s + "+" + AttributeRequirements[s].ToString();

            if (SpellPrerequisites != null)
                foreach (string s in SpellPrerequisites)
                    toString += ", " + s;

            toString += ", " + LevelRequirement.ToString();
            toString += ", " + SpellType.ToString();
            toString += ", " + ActivationCost.ToString();
            toString += ", " + CoolDown.ToString();

            foreach (BaseEffectData e in Effects)
                toString += ", " + e.ToString();

            return toString;
        }

        #endregion
    }
}
```

Also, I need to update the TalentData class in a similar fashion. Replace the TalentData class with the following code.

```
using RpgLibrary.EffectClasses;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace RpgLibrary.TalentClasses
{
    public enum TalentType { Passive, Sustained, Activated }

    public class TalentData
    {
        #region Field Region

        public string Name;
        public string[] AllowedClasses;
        public Dictionary<string, int> AttributeRequirements;
        public string[] TalentPrerequisites;
```

```
public int LevelRequirement;
public TalentType TalentType;
public int ActivationCost;
public double CoolDown;
public List<BaseEffectData> Effects = new List<BaseEffectData>();

#endregion

#region Property Region

public double CastTime { get; set; }
public double Duration { get; set; }
public double Range { get; set; }
public double AreaOfEffect { get; set; }
public double AngleOfEffect { get; set; }

#endregion

#region Constructor Region

public TalentData()
{
    AttributeRequirements = new Dictionary<string, int>();
}

#endregion

#region Method Region
#endregion

#region Virtual Method region

public override string ToString()
{
    string toString = Name;

    foreach (string s in AllowedClasses)
        toString += ", " + s;

    foreach (string s in AttributeRequirements.Keys)
        toString += ", " + s + "+" + AttributeRequirements[s].ToString();

    if (TalentPrerequisites != null)
        foreach (string s in TalentPrerequisites)
            toString += ", " + s;

    toString += ", " + LevelRequirement.ToString();
    toString += ", " + TalentType.ToString();
    toString += ", " + ActivationCost.ToString();
    toString += ", " + CoolDown.ToString();

    foreach (BaseEffectData s in Effects)
        toString += ", " + s.ToString();
```

```
            return toString;
        }

        #endregion
    }
}
```

In both classes, I replaced the List<BaseEffect> with List<BaseEffectData>. Also, I updated the override of the ToString method to use BaseEffectData instead of BaseEffect.

Now, I will update the FormSpellDetails form. I had to redesign it to handle the new fields that were added to the game, such as mana cost, range, and area of effect. Open FormSpellDetails.Designer.cs and replace its contents with the following code.

Again, this is all generated code and not very interesting, so I will skip it. It is just simpler to do it this way than talk you through adding the controls and setting their properties.

```
namespace RpgEditor
{
    partial class FormSpellDetails
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.groupBox1 = new System.Windows.Forms.GroupBox();
            this.rbConstitution = new System.Windows.Forms.RadioButton();
            this.rbMagic = new System.Windows.Forms.RadioButton();
```

```
            this.rbWillpower = new System.Windows.Forms.RadioButton();
            this.rbCunning = new System.Windows.Forms.RadioButton();
            this.rbDexterity = new System.Windows.Forms.RadioButton();
            this.rbStrength = new System.Windows.Forms.RadioButton();
            this.tbName = new System.Windows.Forms.TextBox();
            this.label1 = new System.Windows.Forms.Label();
            this.groupBox2 = new System.Windows.Forms.GroupBox();
            this.btnEdit = new System.Windows.Forms.Button();
            this.btnRemove = new System.Windows.Forms.Button();
            this.btnAdd = new System.Windows.Forms.Button();
            this.lbEffects = new System.Windows.Forms.ListBox();
            this.btnCancel = new System.Windows.Forms.Button();
            this.btnOK = new System.Windows.Forms.Button();
            this.btnRemoveAllowed = new System.Windows.Forms.Button();
            this.btnMoveAllowed = new System.Windows.Forms.Button();
            this.label10 = new System.Windows.Forms.Label();
            this.label9 = new System.Windows.Forms.Label();
            this.lbAllowedClasses = new System.Windows.Forms.ListBox();
            this.lbClasses = new System.Windows.Forms.ListBox();
            this.label2 = new System.Windows.Forms.Label();
            this.sbLevel = new System.Windows.Forms.NumericUpDown();
            this.label3 = new System.Windows.Forms.Label();
            this.sbMana = new System.Windows.Forms.NumericUpDown();
            this.label4 = new System.Windows.Forms.Label();
            this.sbCoolDown = new System.Windows.Forms.NumericUpDown();
            this.label5 = new System.Windows.Forms.Label();
            this.sbAreaOfEffect = new System.Windows.Forms.NumericUpDown();
            this.label6 = new System.Windows.Forms.Label();
            this.rbRange = new System.Windows.Forms.NumericUpDown();
            this.label7 = new System.Windows.Forms.Label();
            this.sbDuration = new System.Windows.Forms.NumericUpDown();
            this.label8 = new System.Windows.Forms.Label();
            this.cboSpellType = new System.Windows.Forms.ComboBox();
            this.groupBox1.SuspendLayout();
            this.groupBox2.SuspendLayout();
            ((System.ComponentModel.ISupportInitialize)(this.sbLevel)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.sbMana)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.sbCoolDown)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)
(this.sbAreaOfEffect)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.rbRange)).BeginInit();
            ((System.ComponentModel.ISupportInitialize)(this.sbDuration)).BeginInit();
            this.SuspendLayout();
            //
            // groupBox1
            //
            this.groupBox1.Controls.Add(this.rbConstitution);
            this.groupBox1.Controls.Add(this.rbMagic);
            this.groupBox1.Controls.Add(this.rbWillpower);
            this.groupBox1.Controls.Add(this.rbCunning);
            this.groupBox1.Controls.Add(this.rbDexterity);
            this.groupBox1.Controls.Add(this.rbStrength);
            this.groupBox1.Location = new System.Drawing.Point(13, 44);
            this.groupBox1.Margin = new System.Windows.Forms.Padding(2);
```

```
this.groupBox1.Name = "groupBox1";
this.groupBox1.Padding = new System.Windows.Forms.Padding(2);
this.groupBox1.Size = new System.Drawing.Size(110, 186);
this.groupBox1.TabIndex = 8;
this.groupBox1.TabStop = false;
this.groupBox1.Text = "Primary Attribute";
//
// rbConstitution
//
this.rbConstitution.AutoSize = true;
this.rbConstitution.Location = new System.Drawing.Point(5, 128);
this.rbConstitution.Margin = new System.Windows.Forms.Padding(2);
this.rbConstitution.Name = "rbConstitution";
this.rbConstitution.Size = new System.Drawing.Size(80, 17);
this.rbConstitution.TabIndex = 5;
this.rbConstitution.Text = "Constitution";
this.rbConstitution.UseVisualStyleBackColor = true;
//
// rbMagic
//
this.rbMagic.AutoSize = true;
this.rbMagic.Location = new System.Drawing.Point(5, 106);
this.rbMagic.Margin = new System.Windows.Forms.Padding(2);
this.rbMagic.Name = "rbMagic";
this.rbMagic.Size = new System.Drawing.Size(54, 17);
this.rbMagic.TabIndex = 4;
this.rbMagic.Text = "Magic";
this.rbMagic.UseVisualStyleBackColor = true;
//
// rbWillpower
//
this.rbWillpower.AutoSize = true;
this.rbWillpower.Location = new System.Drawing.Point(5, 84);
this.rbWillpower.Margin = new System.Windows.Forms.Padding(2);
this.rbWillpower.Name = "rbWillpower";
this.rbWillpower.Size = new System.Drawing.Size(71, 17);
this.rbWillpower.TabIndex = 3;
this.rbWillpower.Text = "Willpower";
this.rbWillpower.UseVisualStyleBackColor = true;
//
// rbCunning
//
this.rbCunning.AutoSize = true;
this.rbCunning.Location = new System.Drawing.Point(5, 63);
this.rbCunning.Margin = new System.Windows.Forms.Padding(2);
this.rbCunning.Name = "rbCunning";
this.rbCunning.Size = new System.Drawing.Size(64, 17);
this.rbCunning.TabIndex = 2;
this.rbCunning.Text = "Cunning";
this.rbCunning.UseVisualStyleBackColor = true;
//
// rbDexterity
//
this.rbDexterity.AutoSize = true;
```

```
this.rbDexterity.Location = new System.Drawing.Point(5, 41);
this.rbDexterity.Margin = new System.Windows.Forms.Padding(2);
this.rbDexterity.Name = "rbDexterity";
this.rbDexterity.Size = new System.Drawing.Size(66, 17);
this.rbDexterity.TabIndex = 1;
this.rbDexterity.Text = "Dexterity";
this.rbDexterity.UseVisualStyleBackColor = true;
//
// rbStrength
//
this.rbStrength.AutoSize = true;
this.rbStrength.Checked = true;
this.rbStrength.Location = new System.Drawing.Point(5, 18);
this.rbStrength.Margin = new System.Windows.Forms.Padding(2);
this.rbStrength.Name = "rbStrength";
this.rbStrength.Size = new System.Drawing.Size(65, 17);
this.rbStrength.TabIndex = 0;
this.rbStrength.TabStop = true;
this.rbStrength.Text = "Strength";
this.rbStrength.UseVisualStyleBackColor = true;
//
// tbName
//
this.tbName.Location = new System.Drawing.Point(76, 14);
this.tbName.Margin = new System.Windows.Forms.Padding(2);
this.tbName.Name = "tbName";
this.tbName.Size = new System.Drawing.Size(113, 20);
this.tbName.TabIndex = 7;
//
// label1
//
this.label1.AutoSize = true;
this.label1.Location = new System.Drawing.Point(13, 18);
this.label1.Margin = new System.Windows.Forms.Padding(2, 0, 2, 0);
this.label1.Name = "label1";
this.label1.Size = new System.Drawing.Size(64, 13);
this.label1.TabIndex = 6;
this.label1.Text = "Spell Name:";
//
// groupBox2
//
this.groupBox2.Controls.Add(this.btnEdit);
this.groupBox2.Controls.Add(this.btnRemove);
this.groupBox2.Controls.Add(this.btnAdd);
this.groupBox2.Controls.Add(this.lbEffects);
this.groupBox2.Location = new System.Drawing.Point(145, 44);
this.groupBox2.Margin = new System.Windows.Forms.Padding(2);
this.groupBox2.Name = "groupBox2";
this.groupBox2.Padding = new System.Windows.Forms.Padding(2);
this.groupBox2.Size = new System.Drawing.Size(310, 186);
this.groupBox2.TabIndex = 9;
this.groupBox2.TabStop = false;
this.groupBox2.Text = "Effects";
//
```

```
            // btnEdit
            //
            this.btnEdit.Location = new System.Drawing.Point(192, 163);
            this.btnEdit.Margin = new System.Windows.Forms.Padding(2);
            this.btnEdit.Name = "btnEdit";
            this.btnEdit.Size = new System.Drawing.Size(56, 19);
            this.btnEdit.TabIndex = 3;
            this.btnEdit.Text = "Edit";
            this.btnEdit.UseVisualStyleBackColor = true;
            //
            // btnRemove
            //
            this.btnRemove.Location = new System.Drawing.Point(131, 163);
            this.btnRemove.Margin = new System.Windows.Forms.Padding(2);
            this.btnRemove.Name = "btnRemove";
            this.btnRemove.Size = new System.Drawing.Size(56, 19);
            this.btnRemove.TabIndex = 2;
            this.btnRemove.Text = "Remove";
            this.btnRemove.UseVisualStyleBackColor = true;
            //
            // btnAdd
            //
            this.btnAdd.Location = new System.Drawing.Point(70, 163);
            this.btnAdd.Margin = new System.Windows.Forms.Padding(2);
            this.btnAdd.Name = "btnAdd";
            this.btnAdd.Size = new System.Drawing.Size(56, 19);
            this.btnAdd.TabIndex = 1;
            this.btnAdd.Text = "Add";
            this.btnAdd.UseVisualStyleBackColor = true;
            //
            // lbEffects
            //
            this.lbEffects.FormattingEnabled = true;
            this.lbEffects.Location = new System.Drawing.Point(22, 17);
            this.lbEffects.Margin = new System.Windows.Forms.Padding(2);
            this.lbEffects.Name = "lbEffects";
            this.lbEffects.Size = new System.Drawing.Size(272, 134);
            this.lbEffects.TabIndex = 0;
            //
            // btnCancel
            //
            this.btnCancel.Anchor = ((System.Windows.Forms.AnchorStyles)
((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Right)));
            this.btnCancel.Location = new System.Drawing.Point(797, 44);
            this.btnCancel.Margin = new System.Windows.Forms.Padding(2);
            this.btnCancel.Name = "btnCancel";
            this.btnCancel.Size = new System.Drawing.Size(56, 19);
            this.btnCancel.TabIndex = 11;
            this.btnCancel.Text = "Cancel";
            this.btnCancel.UseVisualStyleBackColor = true;
            //
            // btnOK
            //
            this.btnOK.Anchor = ((System.Windows.Forms.AnchorStyles)
```

```
((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Right)));
            this.btnOK.Location = new System.Drawing.Point(797, 21);
            this.btnOK.Margin = new System.Windows.Forms.Padding(2);
            this.btnOK.Name = "btnOK";
            this.btnOK.Size = new System.Drawing.Size(56, 19);
            this.btnOK.TabIndex = 10;
            this.btnOK.Text = "OK";
            this.btnOK.UseVisualStyleBackColor = true;
            //
            // btnRemoveAllowed
            //
            this.btnRemoveAllowed.Location = new System.Drawing.Point(585, 126);
            this.btnRemoveAllowed.Name = "btnRemoveAllowed";
            this.btnRemoveAllowed.Size = new System.Drawing.Size(75, 23);
            this.btnRemoveAllowed.TabIndex = 54;
            this.btnRemoveAllowed.Text = "<<";
            this.btnRemoveAllowed.UseVisualStyleBackColor = true;
            //
            // btnMoveAllowed
            //
            this.btnMoveAllowed.Location = new System.Drawing.Point(585, 97);
            this.btnMoveAllowed.Name = "btnMoveAllowed";
            this.btnMoveAllowed.Size = new System.Drawing.Size(75, 23);
            this.btnMoveAllowed.TabIndex = 55;
            this.btnMoveAllowed.Text = ">>";
            this.btnMoveAllowed.UseVisualStyleBackColor = true;
            //
            // label10
            //
            this.label10.AutoSize = true;
            this.label10.Location = new System.Drawing.Point(663, 29);
            this.label10.Name = "label10";
            this.label10.Size = new System.Drawing.Size(86, 13);
            this.label10.TabIndex = 52;
            this.label10.Text = "Allowed Classes:";
            //
            // label9
            //
            this.label9.AutoSize = true;
            this.label9.Location = new System.Drawing.Point(459, 29);
            this.label9.Name = "label9";
            this.label9.Size = new System.Drawing.Size(95, 13);
            this.label9.TabIndex = 53;
            this.label9.Text = "Character Classes:";
            //
            // lbAllowedClasses
            //
            this.lbAllowedClasses.FormattingEnabled = true;
            this.lbAllowedClasses.Location = new System.Drawing.Point(666, 53);
            this.lbAllowedClasses.Name = "lbAllowedClasses";
            this.lbAllowedClasses.Size = new System.Drawing.Size(120, 173);
            this.lbAllowedClasses.TabIndex = 50;
            //
            // lbClasses
```

```
//
this.lbClasses.FormattingEnabled = true;
this.lbClasses.Location = new System.Drawing.Point(459, 53);
this.lbClasses.Name = "lbClasses";
this.lbClasses.Size = new System.Drawing.Size(120, 173);
this.lbClasses.TabIndex = 51;
//
// label2
//
this.label2.AutoSize = true;
this.label2.Location = new System.Drawing.Point(15, 250);
this.label2.Name = "label2";
this.label2.Size = new System.Drawing.Size(36, 13);
this.label2.TabIndex = 56;
this.label2.Text = "Level:";
//
// sbLevel
//
this.sbLevel.Location = new System.Drawing.Point(57, 246);
this.sbLevel.Minimum = new decimal(new int[] {
1,
0,
0,
0});
this.sbLevel.Name = "sbLevel";
this.sbLevel.Size = new System.Drawing.Size(66, 20);
this.sbLevel.TabIndex = 57;
this.sbLevel.Value = new decimal(new int[] {
1,
0,
0,
0});
//
// label3
//
this.label3.AutoSize = true;
this.label3.Location = new System.Drawing.Point(129, 250);
this.label3.Name = "label3";
this.label3.Size = new System.Drawing.Size(61, 13);
this.label3.TabIndex = 58;
this.label3.Text = "Mana Cost:";
//
// sbMana
//
this.sbMana.Location = new System.Drawing.Point(196, 246);
this.sbMana.Minimum = new decimal(new int[] {
1,
0,
0,
0});
this.sbMana.Name = "sbMana";
this.sbMana.Size = new System.Drawing.Size(62, 20);
this.sbMana.TabIndex = 59;
this.sbMana.Value = new decimal(new int[] {
```

```
1,
0,
0,
0});
//
// label4
//
this.label4.AutoSize = true;
this.label4.Location = new System.Drawing.Point(264, 250);
this.label4.Name = "label4";
this.label4.Size = new System.Drawing.Size(62, 13);
this.label4.TabIndex = 60;
this.label4.Text = "Cool Down:";
//
// sbCoolDown
//
this.sbCoolDown.DecimalPlaces = 3;
this.sbCoolDown.Increment = new decimal(new int[] {
5,
0,
0,
196608});
this.sbCoolDown.Location = new System.Drawing.Point(332, 246);
this.sbCoolDown.Name = "sbCoolDown";
this.sbCoolDown.Size = new System.Drawing.Size(69, 20);
this.sbCoolDown.TabIndex = 61;
//
// label5
//
this.label5.AutoSize = true;
this.label5.Location = new System.Drawing.Point(407, 250);
this.label5.Name = "label5";
this.label5.Size = new System.Drawing.Size(72, 13);
this.label5.TabIndex = 62;
this.label5.Text = "Area of Effect";
//
// sbAreaOfEffect
//
this.sbAreaOfEffect.Location = new System.Drawing.Point(485, 246);
this.sbAreaOfEffect.Name = "sbAreaOfEffect";
this.sbAreaOfEffect.Size = new System.Drawing.Size(81, 20);
this.sbAreaOfEffect.TabIndex = 63;
//
// label6
//
this.label6.AutoSize = true;
this.label6.Location = new System.Drawing.Point(582, 250);
this.btnEdit.Name = "btnEdit";
this.btnEdit.Size = new System.Drawing.Size(56, 19);
this.btnEdit.TabIndex = 3;
this.btnEdit.Text = "Edit";
this.btnEdit.UseVisualStyleBackColor = true;
//
// btnRemove
```

```
            //
            this.btnRemove.Location = new System.Drawing.Point(131, 163);
            this.btnRemove.Margin = new System.Windows.Forms.Padding(2);
            this.btnRemove.Name = "btnRemove";
            this.btnRemove.Size = new System.Drawing.Size(56, 19);
            this.btnRemove.TabIndex = 2;
            this.btnRemove.Text = "Remove";
            this.btnRemove.UseVisualStyleBackColor = true;
            //
            // btnAdd
            //
            this.btnAdd.Location = new System.Drawing.Point(70, 163);
            this.btnAdd.Margin = new System.Windows.Forms.Padding(2);
            this.btnAdd.Name = "btnAdd";
            this.btnAdd.Size = new System.Drawing.Size(56, 19);
            this.btnAdd.TabIndex = 1;
            this.btnAdd.Text = "Add";
            this.btnAdd.UseVisualStyleBackColor = true;
            //
            // lbEffects
            //
            this.lbEffects.FormattingEnabled = true;
            this.lbEffects.Location = new System.Drawing.Point(22, 17);
            this.lbEffects.Margin = new System.Windows.Forms.Padding(2);
            this.lbEffects.Name = "lbEffects";
            this.lbEffects.Size = new System.Drawing.Size(272, 134);
            this.lbEffects.TabIndex = 0;
            //
            // btnCancel
            //
            this.btnCancel.Anchor = ((System.Windows.Forms.AnchorStyles)
((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Right)));
            this.btnCancel.Location = new System.Drawing.Point(797, 44);
            this.btnCancel.Margin = new System.Windows.Forms.Padding(2);
            this.btnCancel.Name = "btnCancel";
            this.btnCancel.Size = new System.Drawing.Size(56, 19);
            this.btnCancel.TabIndex = 11;
            this.btnCancel.Text = "Cancel";
            this.btnCancel.UseVisualStyleBackColor = true;
            //
            // btnOK
            //
            this.btnOK.Anchor = ((System.Windows.Forms.AnchorStyles)
((System.Windows.Forms.AnchorStyles.Top | System.Windows.Forms.AnchorStyles.Right)));
            this.btnOK.Location = new System.Drawing.Point(797, 21);
            this.btnOK.Margin = new System.Windows.Forms.Padding(2);
            this.btnOK.Name = "btnOK";
            this.btnOK.Size = new System.Drawing.Size(56, 19);
            this.btnOK.TabIndex = 10;
            this.btnOK.Text = "OK";
            this.btnOK.UseVisualStyleBackColor = true;
            //
            // btnRemoveAllowed
            //
```

```
this.btnRemoveAllowed.Location = new System.Drawing.Point(585, 126);
this.btnRemoveAllowed.Name = "btnRemoveAllowed";
this.btnRemoveAllowed.Size = new System.Drawing.Size(75, 23);
this.btnRemoveAllowed.TabIndex = 54;
this.btnRemoveAllowed.Text = "<<";
this.btnRemoveAllowed.UseVisualStyleBackColor = true;
//
// btnMoveAllowed
//
this.btnMoveAllowed.Location = new System.Drawing.Point(585, 97);
this.btnMoveAllowed.Name = "btnMoveAllowed";
this.btnMoveAllowed.Size = new System.Drawing.Size(75, 23);
this.btnMoveAllowed.TabIndex = 55;
this.btnMoveAllowed.Text = ">>";
this.btnMoveAllowed.UseVisualStyleBackColor = true;
//
// label10
//
this.label10.AutoSize = true;
this.label10.Location = new System.Drawing.Point(663, 29);
this.label10.Name = "label10";
this.label10.Size = new System.Drawing.Size(86, 13);
this.label10.TabIndex = 52;
this.label10.Text = "Allowed Classes:";
//
// label9
//
this.label9.AutoSize = true;
this.label9.Location = new System.Drawing.Point(459, 29);
this.label9.Name = "label9";
this.label9.Size = new System.Drawing.Size(95, 13);
this.label9.TabIndex = 53;
this.label9.Text = "Character Classes:";
//
// lbAllowedClasses
//
this.lbAllowedClasses.FormattingEnabled = true;
this.lbAllowedClasses.Location = new System.Drawing.Point(666, 53);
this.lbAllowedClasses.Name = "lbAllowedClasses";
this.lbAllowedClasses.Size = new System.Drawing.Size(120, 173);
this.lbAllowedClasses.TabIndex = 50;
//
// lbClasses
//
this.lbClasses.FormattingEnabled = true;
this.lbClasses.Location = new System.Drawing.Point(459, 53);
this.lbClasses.Name = "lbClasses";
this.lbClasses.Size = new System.Drawing.Size(120, 173);
this.lbClasses.TabIndex = 51;
//
// label2
//
this.label2.AutoSize = true;
this.label2.Location = new System.Drawing.Point(15, 250);
```

```
this.label2.Name = "label2";
this.label2.Size = new System.Drawing.Size(36, 13);
this.label2.TabIndex = 56;
this.label2.Text = "Level:";
//
// sbLevel
//
this.sbLevel.Location = new System.Drawing.Point(57, 246);
this.sbLevel.Minimum = new decimal(new int[] {
1,
0,
0,
0});
this.sbLevel.Name = "sbLevel";
this.sbLevel.Size = new System.Drawing.Size(66, 20);
this.sbLevel.TabIndex = 57;
this.sbLevel.Value = new decimal(new int[] {
1,
0,
0,
0});
//
// label3
//
this.label3.AutoSize = true;
this.label3.Location = new System.Drawing.Point(129, 250);
this.label3.Name = "label3";
this.label3.Size = new System.Drawing.Size(61, 13);
this.label3.TabIndex = 58;
this.label3.Text = "Mana Cost:";
//
// sbMana
//
this.sbMana.Location = new System.Drawing.Point(196, 246);
this.sbMana.Minimum = new decimal(new int[] {
1,
0,
0,
0});
this.sbMana.Name = "sbMana";
this.sbMana.Size = new System.Drawing.Size(62, 20);
this.sbMana.TabIndex = 59;
this.sbMana.Value = new decimal(new int[] {
1,
0,
0,
0});
//
// label4
//
this.label4.AutoSize = true;
this.label4.Location = new System.Drawing.Point(264, 250);
this.label4.Name = "label4";
this.label4.Size = new System.Drawing.Size(62, 13);
```

```
this.label4.TabIndex = 60;
this.label4.Text = "Cool Down:";
//
// sbCoolDown
//
this.sbCoolDown.DecimalPlaces = 3;
this.sbCoolDown.Increment = new decimal(new int[] {
5,
0,
0,
196608});
this.sbCoolDown.Location = new System.Drawing.Point(332, 246);
this.sbCoolDown.Name = "sbCoolDown";
this.sbCoolDown.Size = new System.Drawing.Size(69, 20);
this.sbCoolDown.TabIndex = 61;
//
// label5
//
this.label5.AutoSize = true;
this.label5.Location = new System.Drawing.Point(407, 250);
this.label5.Name = "label5";
this.label5.Size = new System.Drawing.Size(72, 13);
this.label5.TabIndex = 62;
this.label5.Text = "Area of Effect";
//
// sbAreaOfEffect
//
this.sbAreaOfEffect.Location = new System.Drawing.Point(485, 246);
this.sbAreaOfEffect.Name = "sbAreaOfEffect";
this.sbAreaOfEffect.Size = new System.Drawing.Size(81, 20);
this.sbAreaOfEffect.TabIndex = 63;
//
// label6
//
this.label6.AutoSize = true;
this.label6.Location = new System.Drawing.Point(582, 250);
this.label6.Name = "label6";
this.label6.Size = new System.Drawing.Size(42, 13);
this.label6.TabIndex = 64;
this.label6.Text = "Range:";
//
// rbRange
//
this.rbRange.Location = new System.Drawing.Point(630, 246);
this.rbRange.Name = "rbRange";
this.rbRange.Size = new System.Drawing.Size(65, 20);
this.rbRange.TabIndex = 65;
//
// label7
//
this.label7.AutoSize = true;
this.label7.Location = new System.Drawing.Point(716, 250);
this.label7.Name = "label7";
this.label7.Size = new System.Drawing.Size(50, 13);
```

```
this.label7.TabIndex = 66;
this.label7.Text = "Duration:";
//
// sbDuration
//
this.sbDuration.Location = new System.Drawing.Point(772, 246);
this.sbDuration.Name = "sbDuration";
this.sbDuration.Size = new System.Drawing.Size(59, 20);
this.sbDuration.TabIndex = 67;
//
// label8
//
this.label8.AutoSize = true;
this.label8.Location = new System.Drawing.Point(212, 18);
this.label8.Name = "label8";
this.label8.Size = new System.Drawing.Size(60, 13);
this.label8.TabIndex = 68;
this.label8.Text = "Spell Type:";
//
// cboSpellType
//
this.cboSpellType.FormattingEnabled = true;
this.cboSpellType.Location = new System.Drawing.Point(289, 14);
this.cboSpellType.Name = "cboSpellType";
this.cboSpellType.Size = new System.Drawing.Size(121, 21);
this.cboSpellType.TabIndex = 69;
//
// FormSpellDetails
//
this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
this.ClientSize = new System.Drawing.Size(864, 282);
this.Controls.Add(this.cboSpellType);
this.Controls.Add(this.label8);
this.Controls.Add(this.sbDuration);
this.Controls.Add(this.label7);
this.Controls.Add(this.rbRange);
this.Controls.Add(this.label6);
this.Controls.Add(this.sbAreaOfEffect);
this.Controls.Add(this.label5);
this.Controls.Add(this.sbCoolDown);
this.Controls.Add(this.label4);
this.Controls.Add(this.sbMana);
this.Controls.Add(this.label3);
this.Controls.Add(this.sbLevel);
this.Controls.Add(this.label2);
this.Controls.Add(this.btnRemoveAllowed);
this.Controls.Add(this.btnMoveAllowed);
this.Controls.Add(this.label10);
this.Controls.Add(this.label9);
this.Controls.Add(this.lbAllowedClasses);
this.Controls.Add(this.lbClasses);
this.Controls.Add(this.groupBox1);
this.Controls.Add(this.tbName);
```

```
        this.Controls.Add(this.label1);
        this.Controls.Add(this.groupBox2);
        this.Controls.Add(this.btnCancel);
        this.Controls.Add(this.btnOK);
        this.Name = "FormSpellDetails";
        this.Text = "FormSpellDetails";
        this.Load += new System.EventHandler(this.FormSpellDetails_Load);
        this.groupBox1.ResumeLayout(false);
        this.groupBox1.PerformLayout();
        this.groupBox2.ResumeLayout(false);
        ((System.ComponentModel.ISupportInitialize)(this.sbLevel)).EndInit();
        ((System.ComponentModel.ISupportInitialize)(this.sbMana)).EndInit();
        ((System.ComponentModel.ISupportInitialize)(this.sbCoolDown)).EndInit();
        ((System.ComponentModel.ISupportInitialize)(this.sbAreaOfEffect)).EndInit();
        ((System.ComponentModel.ISupportInitialize)(this.rbRange)).EndInit();
        ((System.ComponentModel.ISupportInitialize)(this.sbDuration)).EndInit();
        this.ResumeLayout(false);
        this.PerformLayout();

    }

    #endregion

    private System.Windows.Forms.GroupBox groupBox1;
    private System.Windows.Forms.RadioButton rbConstitution;
    private System.Windows.Forms.RadioButton rbMagic;
    private System.Windows.Forms.RadioButton rbWillpower;
    private System.Windows.Forms.RadioButton rbCunning;
    private System.Windows.Forms.RadioButton rbDexterity;
    private System.Windows.Forms.RadioButton rbStrength;
    private System.Windows.Forms.TextBox tbName;
    private System.Windows.Forms.Label label1;
    private System.Windows.Forms.GroupBox groupBox2;
    private System.Windows.Forms.Button btnEdit;
    private System.Windows.Forms.Button btnRemove;
    private System.Windows.Forms.Button btnAdd;
    private System.Windows.Forms.ListBox lbEffects;
    private System.Windows.Forms.Button btnCancel;
    private System.Windows.Forms.Button btnOK;
    private System.Windows.Forms.Button btnRemoveAllowed;
    private System.Windows.Forms.Button btnMoveAllowed;
    private System.Windows.Forms.Label label10;
    private System.Windows.Forms.Label label9;
    private System.Windows.Forms.ListBox lbAllowedClasses;
    private System.Windows.Forms.ListBox lbClasses;
    private System.Windows.Forms.Label label2;
    private System.Windows.Forms.NumericUpDown sbLevel;
    private System.Windows.Forms.Label label3;
    private System.Windows.Forms.NumericUpDown sbMana;
    private System.Windows.Forms.Label label4;
    private System.Windows.Forms.NumericUpDown sbCoolDown;
    private System.Windows.Forms.Label label5;
    private System.Windows.Forms.NumericUpDown sbAreaOfEffect;
    private System.Windows.Forms.Label label6;
```

```
        private System.Windows.Forms.NumericUpDown rbRange;
        private System.Windows.Forms.Label label7;
        private System.Windows.Forms.NumericUpDown sbDuration;
        private System.Windows.Forms.Label label8;
        private System.Windows.Forms.ComboBox cboSpellType;
    }
}
```

Now, I will update the FormSpellDetails to call the new form.To begin, click the FormSpellDetails form and press the F7 key to bring up the code. Replace the code for that form with the following code.

```
using RpgLibrary.EffectClasses;
using RpgLibrary.SpellClasses;
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 RpgEditor
{
    public partial class FormSpellDetails : Form
    {
        public SpellData Spell { get; set; }

        public FormSpellDetails()
        {
            InitializeComponent();

            btnAdd.Click += BtnAdd_Click;
            btnRemove.Click += BtnRemove_Click;

            btnOK.Click += BtnOK_Click;
            btnCancel.Click += BtnCancel_Click;

            btnMoveAllowed.Click += BtnMoveAllowed_Click;
            btnRemoveAllowed.Click += BtnRemoveAllowed_Click;
            Spell = new SpellData();
        }

        private void BtnRemoveAllowed_Click(object sender, EventArgs e)
        {
            if (lbAllowedClasses.SelectedItem != null)
            {
                lbClasses.Items.Add(lbAllowedClasses.SelectedItem);
                lbAllowedClasses.Items.RemoveAt(lbAllowedClasses.SelectedIndex);
            }
        }
```

```
        private void BtnMoveAllowed_Click(object sender, EventArgs e)
        {
            if (lbClasses.SelectedItem != null)
            {
                lbAllowedClasses.Items.Add(lbClasses.SelectedItem);
                lbClasses.Items.RemoveAt(lbClasses.SelectedIndex);
            }
        }

        private void BtnCancel_Click(object sender, EventArgs e)
        {
            Spell = null;
            Close();
        }

        private void BtnOK_Click(object sender, EventArgs e)
        {
            if (string.IsNullOrEmpty(tbName.Text))
            {
                MessageBox.Show("The spell must have a name.");
                return;
            }

            if (lbEffects.Items.Count == 0)
            {
                MessageBox.Show("Spell must have at least one effect.");
                return;
            }

            if (lbAllowedClasses.Items.Count == 0)
            {
                MessageBox.Show("Spell mustg belong to at least one class.");
                return;
            }

            Spell.Name = tbName.Text;

            Spell.ActivationCost = (int)sbMana.Value;
            Spell.LevelRequirement = (int)sbLevel.Value;
            Spell.Duration = (float)sbDuration.Value;
            Spell.AreaOfEffect = (int)sbAreaOfEffect.Value;
            Spell.SpellType = (SpellType)Enum.Parse(typeof(SpellType),
cboSpellType.SelectedItem.ToString());
            Spell.AllowedClasses = new string[lbAllowedClasses.Items.Count];

            for (int i = 0; i < lbAllowedClasses.Items.Count; i++)
            {
                Spell.AllowedClasses[i] = lbAllowedClasses.Items[i].ToString();
            }

            Close();
        }

        private void BtnRemove_Click(object sender, EventArgs e)
```

```
        {
            if (lbEffects.SelectedIndex >= 0)
            {
                Spell.Effects.RemoveAt(lbEffects.SelectedIndex);
                lbEffects.Items.RemoveAt(lbEffects.SelectedIndex);
            }
        }

        private void BtnAdd_Click(object sender, EventArgs e)
        {
            FormBaseEffect frm = new FormBaseEffect();

            frm.ShowDialog();

            if (frm.BaseEffectData != null)
            {
                lbEffects.Items.Add(frm.BaseEffectData.ToString());
                Spell.Effects.Add(frm.BaseEffectData);
            }
        }

        private void FormSpellDetails_Load(object sender, EventArgs e)
        {
            foreach (string s in FormDetails.EntityDataManager.EntityData.Keys)
            {
                lbClasses.Items.Add(s);
            }

            foreach (var v in Enum.GetNames(typeof(SpellType)))
            {
                cboSpellType.Items.Add(v.ToString());
            }
        }
    }
}
```

There are using statements to bring the effect and spell classes into scope. There is a SpellData property that will hold the spell that you have created, or edited in a future tutorial. I wire the event handlers for the Add and Remove buttons to add and remove effects. Again, I am not doing Update operations, due to the length of the tutorial already. I also wire event handers for the OK and Cancel buttons. Finally, I wire handlers for the MoveAllowed and RemoveAllowed buttons that are use to determine if a class is able to use the spell. I also create a new SpellData object.

The event handler for the Click event of btnRemoveAllowed check to see if there is a selected item. If there is it adds the selected item to the list of classes and removes it from the list of allowed classes. The Click event handler for BtnMoveAllowed works in reverse.

The event handler for the Click event of btnCancel is trivial. It sets the Spell property to null and closes the form.

The event handler for the OK button is a little more complicated. Because I used spin boxes for the

numeric values they do not need to be validated. I went with the minimum and maximum properties of the spin boxes. Everything is capped at 100 for the maximum, or zero for the minimum, except for the one that represents level of the character. It is limited to one because there are no zero level characters. I do three validations, though. First, I make sure that the spell has a name. If it doesn't I exit the method. I then make sure that the spell has at least one effect associated with it. The final validation is that the spell is assigned to a class.

If all of the validations pass, I set the property of the spell to the properties of the associated controls. The only interesting part is for the allowed class. For that, I look over the items in the allowed classes list box. They are then set to corresponding entry in the AllowedClasses array. Finally, I close the form.

The event hander for the Remove button is pretty straight forward. It check to see if the SelectedIndex property of the Effects list box is greater than or equal to zero. If it is, it calls RemoveAt on the Effects collection of the spell. Then, it calls the RemoveAt method of Items collection of the list box.

In the event handler for the Add button I create a new FormBaseEffect and call its ShowDialog method. If the BaseEffectData field is not null I add it to the list box.

```
        private void BtnAdd_Click(object sender, EventArgs e)
        {
            FormBaseEffect frm = new FormBaseEffect();

            frm.ShowDialog();

            if (frm.BaseEffectData != null)
            {
                lbEffects.Items.Add(frm.BaseEffectData.ToString());
                Spell.Effects.Add(frm.BaseEffectData);
            }
        }

        private void FormSpellDetails_Load(object sender, EventArgs e)
        {
            foreach (string s in FormDetails.EntityDataManager.EntityData.Keys)
            {
                lbClasses.Items.Add(s);
            }

            foreach (var v in Enum.GetNames(typeof(SpellType)))
            {
                cboSpellType.Items.Add(v.ToString());
            }
        }
    }
}
```

All of these changes broke the SpellDataManager class. It is because SpellData was expecting a BaseEffect but now is expecting a BaseEffectData. It is a trivial update. Just change the local variables to type BaseEffectData and call the right constructor. Replace the SpellDataManager with the

following code.

```
using Microsoft.Xna.Framework;
using RpgLibrary.EffectClasses;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace RpgLibrary.SpellClasses
{
    public class SpellDataManager
    {
        #region Field Region

        readonly Dictionary<string, SpellData> spellData;

        #endregion

        #region Property Region

        public Dictionary<string, SpellData> SpellData
        {
            get { return spellData; }
        }

        public object EffectData { get; private set; }

        #endregion

        #region Constructor Region

        public SpellDataManager()
        {
            spellData = new Dictionary<string, SpellData>();
        }

        #endregion

        #region Method Region

        public void FillSpellData()
        {
            SpellData data = new SpellData()
            {
                Name = "Spark Jolt",
                SpellPrerequisites = new string[0],
                SpellType = SpellType.Activated,
                LevelRequirement = 1,
                ActivationCost = 8,
                AllowedClasses = new string[] { "Wizard" },
                AttributeRequirements = new Dictionary<string, int>() { { "Magic",
10 } },
```

```
            AreaOfEffect = 32,
            AngleOfEffect = MathHelper.TwoPi,
            CastTime = 0,
            Duration = 0,
            CoolDown = 2,
            Range = 128,
        };

        BaseEffectData effect = (new DamageEffectData
        {
            Name = "Spark Jolt",
            TargetType = TargetType.Enemy,
            AttackType = AttackType.Health,
            DamageType = DamageType.Air,
            DieType = DieType.D6,
            NumberOfDice = 3,
            Modifier = 2
        });

        data.Effects.Add(effect);

        spellData.Add("Spark Jolt", data);

        data = new SpellData()
        {
            Name = "Mend",
            SpellPrerequisites = new string[0],
            SpellType = SpellType.Activated,
            LevelRequirement = 1,
            ActivationCost = 6,
            AllowedClasses = new string[] { "Priest" },
            AttributeRequirements = new Dictionary<string, int>() { { "Magic",
10 } },
            Range = 0,
            AreaOfEffect = 0,
            AngleOfEffect = 0,
            CastTime = 0,
            CoolDown = 2
        };

        BaseEffectData healEffect = (new HealEffectData
        {
            Name = "Mend",
            TargetType = TargetType.Self,
            HealType = HealType.Health,
            DieType = DieType.D8,
            NumberOfDice = 2,
            Modifier = 2
        });

        data.Effects.Add(healEffect);
        spellData.Add("Mend", data);
    }
```

```
        #endregion

        #region Virtual Method region
        #endregion
    }
}
```

Similarly, the TalentDataManager class is broken, for the same reasons. Replace that class with this version.

```
using Microsoft.Xna.Framework;
using RpgLibrary.EffectClasses;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace RpgLibrary.TalentClasses
{
    public class TalentDataManager
    {
        #region Field Region

        readonly Dictionary<string, TalentData> talentData;

        #endregion

        #region Property Region

        public Dictionary<string, TalentData> TalentData
        {
            get { return talentData; }
        }

        #endregion

        #region Constructor Region

        public TalentDataManager()
        {
            talentData = new Dictionary<string, TalentData>();
        }

        #endregion

        #region Method Region

        public void FillTalents()
        {
            TalentData data = new TalentData
            {
                Name = "Bash",
                TalentPrerequisites = new string[0],
```

```
                LevelRequirement = 1,
                TalentType = TalentType.Activated,
                ActivationCost = 5,
                AllowedClasses = new string[] { "Fighter" },
                AttributeRequirements = new Dictionary<string, int>() { { "Strength",
10 } },
                AreaOfEffect = 8,
                AngleOfEffect = MathHelper.TwoPi,
                Range = 48,
                CastTime = 0,
                CoolDown = 1.5
            };

            BaseEffectData effect = (new DamageEffectData
            {
                TargetType = SpellClasses.TargetType.Enemy,
                AttackType = AttackType.Health,
                DamageType = DamageType.Crushing,
                DieType = DieType.D8,
                Modifier = 2,
                Name = "Bash",
                NumberOfDice = 2
            });

            data.Effects.Add(effect);

            talentData.Add("Bash", data);

            data = new TalentData()
            {
                Name = "Below The Belt",
                TalentPrerequisites = new string[0],
                LevelRequirement = 1,
                TalentType = TalentType.Activated,
                ActivationCost = 5,
                AllowedClasses = new string[] { "Rogue" },
                AttributeRequirements = new Dictionary<string, int>() { { "Dexterity", 10
} },
                AreaOfEffect = 8,
                AngleOfEffect = MathHelper.TwoPi,
                Range = 48,
                CastTime = 0,
                CoolDown = 1.5
            };

            effect = (new DamageEffectData()
            {
                TargetType = SpellClasses.TargetType.Enemy,
                AttackType = AttackType.Health,
                DamageType = DamageType.Piercing,
                DieType = DieType.D4,
                Modifier = 2,
                Name = "Below The Belt",
                NumberOfDice = 3
```

```
            });

            data.Effects.Add(effect);

            talentData.Add("Below The Belt", data);
        }

        #endregion

        #region Virtual Method region
        #endregion
    }
}
```

Also, the FromData methods of the Spell and Talent classes are broken. Replace the FromSpellData method with the following version.

```
public static Spell FromSpellData(SpellData data)
{
    Spell spell = new Spell
    {
        name = data.Name,
        levelRequirement = data.LevelRequirement,
        spellType = data.SpellType,
        activationCost = data.ActivationCost,
        coolDown = data.CoolDown,
        Range = data.Range,
        AreaOfEffect = data.AreaOfEffect,
        AngleOfEffect = data.AngleOfEffect,
        CastTime = data.CastTime
    };

    foreach (string s in data.AllowedClasses)
        spell.allowedClasses.Add(s.ToLower());

    foreach (string s in data.AttributeRequirements.Keys)
        spell.attributeRequirements.Add(
        s.ToLower(),
        data.AttributeRequirements[s]);

    foreach (string s in data.SpellPrerequisites)
        spell.SpellPrerequisites.Add(s);

    foreach (BaseEffectData s in data.Effects)
    {
        if (s is DamageEffectData damage)
        {
            spell.Effects.Add(DamageEffect.FromDamageEffectData(damage));
        }

        if (s is HealEffectData heal)
        {
            spell.Effects.Add(HealEffect.FromHealEffectData(heal));
```

```
        }
    }

    return spell;
}
```

The change is instead of looping over BaseEffects, I am looping over BaseEffectData objects. Inside the loop I check to see if the effect data is a DamageEffectData and if it is cast it to a local variable. I then create a new DamageEffect using the static method FromDamageEffectData. I follow the same workflow for HealEffectData object.

As I mentioned, I also change the similar method of the Talent class. Replace the FromTalentData method of the Talent class with this new version.

```
public static Talent FromTalentData(TalentData data)
{
    Talent talent = new Talent
    {
        name = data.Name,
        levelRequirement = data.LevelRequirement,
        talentType = data.TalentType,
        activationCost = data.ActivationCost,
        coolDown = data.CoolDown,
        Range = data.Range,
        AreaOfEffect = data.AreaOfEffect,
        AngleOfEffect = data.AngleOfEffect,
        CastTime = data.CastTime
    };

    foreach (string s in data.AllowedClasses)
        talent.allowedClasses.Add(s.ToLower());

    foreach (string s in data.AttributeRequirements.Keys)
        talent.attributeRequirements.Add(
            s.ToLower(),
            data.AttributeRequirements[s]);

    foreach (string s in data.TalentPrerequisites)
        talent.talentPrerequisites.Add(s);

    foreach (BaseEffectData s in data.Effects)
    {
        if (s is DamageEffectData damage)
        {
            talent.Effects.Add(DamageEffect.FromDamageEffectData(damage));
        }

        if (s is HealEffectData heal)
        {
            talent.Effects.Add(HealEffect.FromHealEffectData(heal));
        }
    }
```

```
    return talent;
}
```

So, now I am going to turn my attention to bringing up the spell form. This is a relatively straight forward change to the form so I am going to walk you through it rather than regurgitate all of the system generated code for a few lines. To start, open FormMain in the RpgLibrary project. In the menu bar, click beside the Skills entry. In that space enter S&pells. Click the new entry and set the Name property of the menu item to spellsToolStripMenuItem. Now, bring up the code view by pressing F7. Add the following field and property. Also, replace the constructor to the following code.

```
FormSpell frmSpell;

public static string SpellPath { get; private set; }

public FormMain()
{
    InitializeComponent();

    this.FormClosing += new FormClosingEventHandler(FormMain_FormClosing);

    newGameToolStripMenuItem.Click += new EventHandler(newGameToolStripMenuItem_Click);
    openGameToolStripMenuItem.Click += new EventHandler(openGameToolStripMenuItem_Click);
    saveGameToolStripMenuItem.Click += new EventHandler(saveGameToolStripMenuItem_Click);
    exitToolStripMenuItem.Click += new EventHandler(exitToolStripMenuItem_Click);
    classesToolStripMenuItem.Click += new EventHandler(classesToolStripMenuItem_Click);
    armorToolStripMenuItem.Click += new EventHandler(armorToolStripMenuItem_Click);
    shieldToolStripMenuItem.Click += new EventHandler(shieldToolStripMenuItem_Click);
    weaponToolStripMenuItem.Click += new EventHandler(weaponToolStripMenuItem_Click);
    keysToolStripMenuItem.Click += new EventHandler(keysToolStripMenuItem_Click);
    chestsToolStripMenuItem.Click += new EventHandler(chestsToolStripMenuItem_Click);
    skillsToolStripMenuItem.Click += new EventHandler(skillsToolStripMenuItem_Click);
    spellsToolStripMenuItem.Click += SpellsToolStripMenuItem_Click;
}
```

I just created a field for the form and a property to expose the path to spells. In the constructor I wire an event handler for the click event of the menu item.
There are a few other changes that need to happen to FormMain, while we have it open. First, we need to update the OpenGame method and the method. What I did was set the path to spells in the data folder. Then, I create a new instance of FormSpell and call FillListBox on it to fill the data. Finally, I enable the menu item.

```
private void OpenGame(string path)
{
    gamePath = Path.Combine(path, "Game");
    classPath = Path.Combine(gamePath, "Classes");
    itemPath = Path.Combine(gamePath, "Items");
    keyPath = Path.Combine(gamePath, "Keys");
    chestPath = Path.Combine(gamePath, "Chests");
    skillPath = Path.Combine(gamePath, "Skills");
    SpellPath = Path.Combine(gamePath, "Spells");
```

```
    if (!Directory.Exists(keyPath))
    {
        Directory.CreateDirectory(keyPath);
    }

    if (!Directory.Exists(chestPath))
    {
        Directory.CreateDirectory(chestPath);
    }

    if (!Directory.Exists(skillPath))
    {
        Directory.CreateDirectory(skillPath);
    }

    rolePlayingGame = XnaSerializer.Deserialize<RolePlayingGame>(
        gamePath + @"\Game.xml");

    FormDetails.ReadEntityData();
    FormDetails.ReadItemData();
    FormDetails.ReadKeyData();
    FormDetails.ReadChestData();
    FormDetails.ReadSkillData();

    PrepareForms();
}

private void PrepareForms()
{
    if (frmClasses == null)
    {
        frmClasses = new FormClasses();
        frmClasses.MdiParent = this;
    }

    frmClasses.FillListBox();

    if (frmArmor == null)
    {
        frmArmor = new FormArmor();
        frmArmor.MdiParent = this;
    }

    frmArmor.FillListBox();

    if (frmShield == null)
    {
        frmShield = new FormShield();
        frmShield.MdiParent = this;
    }

    frmShield.FillListBox();
```

```
    if (frmWeapon == null)
    {
        frmWeapon = new FormWeapon();
        frmWeapon.MdiParent = this;
    }

    frmWeapon.FillListBox();

    if (frmKey == null)
    {
        frmKey = new FormKey();
        frmKey.MdiParent = this;
    }

    frmKey.FillListBox();

    if (frmChest == null)
    {
        frmChest = new FormChest();
        frmChest.MdiParent = this;
    }

    frmChest.FillListBox();

    if (frmSkill == null)
    {
        frmSkill = new FormSkill();
        frmSkill.MdiParent = this;
    }

    frmSkill.FillListBox();

    if (frmSpell == null)
    {
        frmSpell = new FormSpell();
        frmSpell.MdiParent = this;
    }

    frmSpell.FillListBox();

    classesToolStripMenuItem.Enabled = true;
    itemsToolStripMenuItem.Enabled = true;
    keysToolStripMenuItem.Enabled = true;
    chestsToolStripMenuItem.Enabled = true;
    skillsToolStripMenuItem.Enabled = true;
    spellToolStripMenuItem.Enabled = true;
}
```

I also needed to update the event handler for new games. Replace that event handler with the following code.

```
void newGameToolStripMenuItem_Click(object sender, EventArgs e)
{
```

```
using (FormNewGame frmNewGame = new FormNewGame())
{
    DialogResult result = frmNewGame.ShowDialog();

    if (result == DialogResult.OK && frmNewGame.RolePlayingGame != null)
    {
        FolderBrowserDialog folderDialog = new FolderBrowserDialog();

        folderDialog.Description = "Select folder to create game in.";
        folderDialog.SelectedPath = Application.StartupPath;

        DialogResult folderResult = folderDialog.ShowDialog();

        if (folderResult == DialogResult.OK)
        {
            try
            {
                gamePath = Path.Combine(folderDialog.SelectedPath, "Game");
                classPath = Path.Combine(gamePath, "Classes");
                itemPath = Path.Combine(gamePath, "Items");
                keyPath = Path.Combine(gamePath, "Keys");
                chestPath = Path.Combine(gamePath, "Chests");
                skillPath = Path.Combine(gamePath, "Skills");
                SpellPath = Path.Combine(gamePath, "Spells");

                if (Directory.Exists(gamePath))
                    throw new Exception("Selected directory already exists.");

                Directory.CreateDirectory(gamePath);
                Directory.CreateDirectory(classPath);
                Directory.CreateDirectory(itemPath + @"\Armor");
                Directory.CreateDirectory(itemPath + @"\Shield");
                Directory.CreateDirectory(itemPath + @"\Weapon");
                Directory.CreateDirectory(keyPath);
                Directory.CreateDirectory(chestPath);
                Directory.CreateDirectory(skillPath);
                Directory.CreateDirectory(SpellPath);

                rolePlayingGame = frmNewGame.RolePlayingGame;

                XnaSerializer.Serialize<RolePlayingGame>(gamePath + @"\Game.xml",
                    rolePlayingGame);
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.ToString());
                return;
            }

            classesToolStripMenuItem.Enabled = true;
            itemsToolStripMenuItem.Enabled = true;
            keysToolStripMenuItem.Enabled = true;
            chestsToolStripMenuItem.Enabled = true;
            skillsToolStripMenuItem.Enabled = true;
```

```
                spellsToolStripMenuItem.Enabled = true;
            }
        }
    }
}
```

All I did was combine a path to a new subfolder, Spells. I then call the CreateDirectory method if the folder does not exist. Finally, I set the Enabled property of the new menu item to true.

This is getting exceedingly long, so I am going to skip saving and loading spells in the editor. I will wire bringing up the spells form, then bringing up the details form from the spell list form. First, add the SpellsToolStripMenuItem_Click method.

```
private void SpellsToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (frmSpell == null)
    {
        frmSpell = new FormSpell
        {
            MdiParent = this
        };
    }

    frmSpell.Show();
    frmSpell.BringToFront();
}
```

This method is like all of the other content items that we wired event handlers for. I check to see if the form is null. If it is, I create a new instance and set the MdiParent property to the form. I then call the Show method to display the form and BringToFront to make it the top most form.

We are getting close, but still a bit to go. The next step is to update FormDetails to have a spell data manager. Add the following property to FormDetails and update the constructor to the following.

```
public static SpellDataManager SpellDataManager
{
    get; protected set;
}

public FormDetails()
{
    InitializeComponent();

    if (FormDetails.ItemManager == null)
        ItemManager = new ItemDataManager();

    if (FormDetails.EntityDataManager == null)
        EntityDataManager = new EntityDataManager();
    this.FormClosing += new FormClosingEventHandler(FormDetails_FormClosing);

    if (SpellDataManager == null)
```

```
    {
        SpellDataManager = new SpellDataManager();
    }
}
```

All I did was add property, rather than a field and a property, for a SpellDataManager that will hold the spells. Then, in the constructor, I check to see if the spell data manager is null. If it is, I create a new one.

Now, go to SpellForm and bring up the code view by pressing F7. Replace the code of that form with the following.

```
using RpgLibrary.SpellClasses;
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 RpgEditor
{
    public partial class FormSpell : FormDetails
    {
        public FormSpell()
        {
            InitializeComponent();

            btnAdd.Click += BtnAdd_Click;
            btnEdit.Click += BtnEdit_Click;
            btnDelete.Click += BtnDelete_Click;
        }

        private void BtnAdd_Click(object sender, EventArgs e)
        {
            FormSpellDetails frmSpell = new FormSpellDetails();
            frmSpell.ShowDialog();

            if (frmSpell.Spell != null)
            {
                AddSpell(frmSpell.Spell);
            }
        }

        private void BtnEdit_Click(object sender, EventArgs e)
        {
        }

        private void BtnDelete_Click(object sender, EventArgs e)
```

```
        {
            if (lbDetails.SelectedItem != null)
            {
                string detail = (string)lbDetails.SelectedItem;
                string[] parts = detail.Split(',');
                string entity = parts[0].Trim();

                DialogResult result = MessageBox.Show(
                    "Are you sure you want to delete " + entity + "?",
                    "Delete",
                    MessageBoxButtons.YesNo);

                if (result == DialogResult.Yes)
                {
                    lbDetails.Items.RemoveAt(lbDetails.SelectedIndex);
                    SpellDataManager.SpellData.Remove(entity);
                }
            }
        }

        private void AddSpell(SpellData spellData)
        {
            if (FormDetails.SpellDataManager.SpellData.ContainsKey(spellData.Name))
            {
                DialogResult result = MessageBox.Show(
                    spellData.Name + " already exists. Overwrite it?",
                    "Existing spell",
                    MessageBoxButtons.YesNo);

                if (result == DialogResult.No)
                    return;

                SpellDataManager.SpellData[spellData.Name] = spellData;
                FillListBox();

                return;
            }

            SpellDataManager.SpellData.Add(spellData.Name, spellData);
            lbDetails.Items.Add(spellData);
        }

        public void FillListBox()
        {
            lbDetails.Items.Clear();

            foreach (string s in FormDetails.SpellDataManager.SpellData.Keys)
                lbDetails.Items.Add(FormDetails.SpellDataManager.SpellData[s]);
        }
    }
}
```

This is similar to the other list forms that we created. There is a using statement to bring the spell

classes into scope. It inherits from FormDetails to have the same layout as the other data forms. In the constructor I wire the event handlers for the Add, Edit and Delete buttons. I currently do not implement the edit feature. I will do that in a future tutorial.

In the event handler for the Add button I create a FormSpellDetails and call its ShowDialog method to display the dialog. If the Spell property of the resulting form is not null I call a method AddSpell that adds the spell to the list box and the data manager for spells.

In the event handler for the Delete button I check to see if the SelectedItem in the list box is not null. If it is I get the text for the line and break it into its parts on the comma. I then display a message box asking if the user wants to delete the spell. If they do, I remove the spell from the list box and the spell data manager.

The AddSpell method checks to see if there is already a spell by that name. If there is I display a message box asking if you want to overwrite the existing spell. If the answer is No I exit the method. Otherwise, I replace the existing spell in the data manager, call the FillListBox method and exit the method. If it is a new spell, I add it to the data manager and to the list box.

The FillListBox method clears the list box of all items. It then loops over all of the keys in the data manager and adds them to the list box.

Wow! This tutorial is ginormous. I am going to end it here. I've covered most of what I had planned on, and I don't want to get into something new at this point. So, please continue to visit my blog, https://cynthiamcmahon.ca/blog/, for the latest news on my tutorials and other goodness.

Good luck with your Game Programming Adventures!

Cynthia