

Shadow Monsters – MonoGame Tutorial Series

Chapter 15

Editor – Part Three

This tutorial series is about creating a Pokemon style game with the MonoGame Framework called Shadow Monsters. The tutorials will make more sense if You read them in order as each tutorial builds on the previous tutorials. You can find the list of tutorials on my blog: [Shadow Monsters](#). The source code for each tutorial will be available as well. I will be using Visual Studio 2019 Community for the series. The code should compile on the 2013, 2015 and 2017 versions as well.

I want to mention though that the series is released as Creative Commons 3.0 Attribution. It means that You are free to use any of the code or graphics in Your own game, even for commercial use, with attribution. Just give credit to Cynthia McMahon and add a link to my site, <https://mygameprogrammingadventures.blogspot.com>. Screenshots of Your project and/or a video of game play would be appreciated.

I also want to mention that I assume You have a basic understanding of C# and MonoGame. If You don't I recommend that You learn basic C# and work with MonoGame a little. Enough to know the basics of fields, properties, methods, classes and the MonoGame framework.

In this tutorial we will be continuing with the editor. The first thing that I want to add is code that will scroll the map. The map will scroll if the arrow keys are pressed or the mouse is over the first or last tile of the view port. What I did was add a method to the Editor class and call it from the Update method. I will save changes to the Update method until the end. Add the following method to the Editor class.

```
private void HandleScrollMap()
{
    if (map == null)
    {
        return;
    }

    if (Xin.MouseAsPoint.X > 64 * 17 && Xin.MouseAsPoint.X < 64 * 18)
    {
        camera.Position = new Vector2(camera.Position.X + 8, camera.Position.Y);
    }

    if (Xin.KeyboardState.IsKeyDown(Keys.Right))
    {
        camera.Position = new Vector2(camera.Position.X + 8, camera.Position.Y);
    }

    if (Xin.MouseAsPoint.Y > 1080 - 64 && Xin.MouseAsPoint.Y < 1080)
    {
        camera.Position = new Vector2(camera.Position.X, camera.Position.Y + 8);
    }

    if (Xin.KeyboardState.IsKeyDown(Keys.Down))
    {

```

```

        camera.Position = new Vector2(camera.Position.X, camera.Position.Y + 8);
    }

    if (Xin.MouseAsPoint.X < 64)
    {
        camera.Position = new Vector2(camera.Position.X - 8, camera.Position.Y);
    }

    if (Xin.KeyboardState.IsKeyDown(Keys.Left))
    {
        camera.Position = new Vector2(camera.Position.X - 8, camera.Position.Y);
    }

    if (Xin.MouseAsPoint.Y < 64)
    {
        camera.Position = new Vector2(camera.Position.X, camera.Position.Y - 8);
    }

    if (Xin.KeyboardState.IsKeyDown(Keys.Up))
    {
        camera.Position = new Vector2(camera.Position.X, camera.Position.Y - 8);
    }

    camera.LockCamera(map, viewPort);
}

```

The first that the method does is check if the map is null. If it is it exits the method. If the X coordinate of the mouse is between the 17th and 18th tile. If it is I move the camera eight pixels right. Next I check if the right arrow key is down. If it is I move the camera eight pixels to the right. Then I check if the Y coordinate of the mouse is between the height of the window and 64 pixels less. If that is true I move the camera eight pixels down and lock the camera. I do the same if the down arrow key is down. If the X coordinate of the mouse is less than 64 I move the camera eight pixels left. I do the same if the left arrow key is down. I check if the Y coordinate of the mouse is less than 64 and if it is move the camera eight pixels up. I do the same if the up arrow key is down. Finally I lock the camera.

I will start by adding creating characters to the editor. Before that I want to make a change to the Character class and the CharacterLayer class. The changes to the Character class were extensive so I will give You the code for the entire class.

```

using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Content;
using Microsoft.Xna.Framework.Graphics;
using ShadowMonsters.ShadowMonsters;
using ShadowMonsters.TileEngine;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;

namespace ShadowMonsters.Characters
{
    public class Character
    {
        #region Constant

```

```

public const float SpeakingRadius = 40f;
public const int MonsterLimit = 6;

#endregion

#region Field Region

protected string name;
protected string textureName;
protected ShadowMonster[] monsters = new ShadowMonster[MonsterLimit];
protected int currentMonster;
protected ShadowMonster givingMonster;
protected AnimatedSprite sprite;
protected Point sourceTile;

protected string conversation;

protected static Game gameRef;

#endregion

#region Property Region

public string Name
{
    get { return name; }
    set { name = value; }
}

public string SpriteName
{
    get { return textureName; }
}
public AnimatedSprite Sprite
{
    get { return sprite; }
}

public ShadowMonster BattleMonster
{
    get { return monsters[currentMonster]; }
}

public ShadowMonster GiveMonster
{
    get { return givingMonster; }
}

public string Conversation
{
    get { return conversation; }
}

public bool Battled
{
    get;
    set;
}

public Point SourceTile

```

```

    {
        get { return sourceTile; }
        set { sourceTile = value; }
    }
    public List<ShadowMonster> BattleMonsters => monsters.ToList<ShadowMonster>();

#endregion

#region Constructor Region

protected Character()
{
}

#endregion

#region Method Region

private static void BuildAnimations()
{
}

public bool NextMonster()
{
    currentMonster++;

    return currentMonster < MonsterLimit && monsters[currentMonster] != null;
}

public static Character FromString(Game game, string characterString)
{
    if (gameRef == null)
    {
        gameRef = game;
    }

    Character character = new Character();
    string[] parts = characterString.Split(',');

    character.name = parts[0];
    character.textureName = parts[1];
    character.sprite = new AnimatedSprite(
        game.Content.Load<Texture2D>(@"CharacterSprites\" + parts[1]),
        Game1.Animations)
    {
        CurrentAnimation = (AnimationKey)Enum.Parse(typeof(AnimationKey), parts[2])
    };
    character.conversation = parts[3];
    character.currentMonster = int.Parse(parts[4]);
    string[] items = parts[5].Split(':');
    character.SourceTile = new Point(int.Parse(items[0]), int.Parse(items[1]));
    character.sprite.Position = new Vector2(
        character.SourceTile.X * Engine.TileWidth,
        character.SourceTile.Y * Engine.TileHeight);

    for (int i = 6; i < 12 && i < parts.Length - 1; i++)
    {
        ShadowMonster monster =
ShadowMonsterManager.GetShadowMonster(parts[i].ToLowerInvariant());
        character.monsters[i - 6] = monster;
    }
}

```

```

    }

    character.givingMonster = ShadowMonsterManager.GetShadowMonster(parts[parts.Length
- 1].ToLowerInvariant());
    return character;
}

public void ChangeMonster(int index)
{
    if (index < 0 || index >= MonsterLimit)
    {
        currentMonster = index;
    }
}

public void SetConversation(string newConversation)
{
    this.conversation = newConversation;
}

public void Update(GameTime gameTime)
{
    sprite.Update(gameTime);
}

public void Draw(GameTime gameTime, SpriteBatch spriteBatch)
{
    sprite.Draw(gameTime, spriteBatch);
}

#endregion

public bool Alive()
{
    for (int i = 0; i < MonsterLimit; i++)
    {
        if (BattleMonsters[i] != null && BattleMonsters[i].CurrentHealth > 0)
        {
            return true;
        }
    }

    return false;
}

public virtual bool Save(BinaryWriter writer)
{
    StringBuilder b = new StringBuilder();

    b.Append(name);
    b.Append(",");
    b.Append(textureName);
    b.Append(",");
    b.Append(sprite.CurrentAnimation);
    b.Append(",");
    b.Append(conversation);
    b.Append(",");
    b.Append(currentMonster);
    b.Append(",");
    b.Append(Battled);
    b.Append(",");

```

```

        b.Append(SourceTile.X + ":" + SourceTile.Y);

        string s = b.ToString();

        writer.Write(s);
        writer.Write(-1);

        foreach (ShadowMonster a in monsters)
        {
            if (a != null)
            {
                a.Save(writer);
                writer.Write(-1);
            }
            else
            {
                writer.Write("*");
                writer.Write(-1);
            }
        }

        if (givingMonster != null)
        {
            givingMonster.Save(writer);
            writer.Write(-1);
        }
        else
        {
            writer.Write("*");
            writer.Write(-1);
        }

        return true;
    }

    public static Character Load(ContentManager content, BinaryReader reader)
    {
        Character c = new Character();

        string data = reader.ReadString();
        string[] parts = data.Split(',');
        c.name = parts[0];
        c.textureName = parts[1];
        c.sprite = new AnimatedSprite(
            content.Load<Texture2D>(
                @"CharacterSprites\" + parts[1]),
            Game1.Animations);
        c.sprite.CurrentAnimation = (AnimationKey)Enum.Parse(typeof(AnimationKey),
parts[2]);
        c.conversation = parts[3];
        c.currentMonster = int.Parse(parts[4]);
        c.Battled = bool.Parse(parts[5]);
        string[] items = parts[6].Split(':');
        c.SourceTile = new Point(int.Parse(items[0]), int.Parse(items[1]));
        c.sprite.Position = new Vector2(
            c.SourceTile.X * Engine.TileWidth,
            c.SourceTile.Y * Engine.TileHeight);

        reader.ReadInt32();
    }

```

```

        for (int i = 0; i < 6; i++)
        {
            string ShadowMonster = reader.ReadString();

            if (ShadowMonster != "")
            {
                c.monsters[i] = ShadowMonster.Load(content, ShadowMonster);
            }

            reader.ReadInt32();
        }

        string giving = reader.ReadString();

        if (giving != "")
        {
            c.givingMonster = ShadowMonster.Load(content, giving);
        }

        reader.ReadInt32();

        return c;
    }
}

```

I wanted to add a source tile field to the Character class. To do that I added a Point field called sourceTile. I then added a read/write property SourceTile. I changed the string to include a colon separated string where the X is on the left of the colon on the Y is on the right. I updated the FromString method to handle the Point. It splits the point on the colon then parses the parts to create a Point. I also shifted the shadow monsters from 5 and 11 to 6 and 12. I also set the position of the sprite to be the source tile. I do that by taking the point and multiplying the X by Engine.TileWidth and Y by Engine.TileHeight. The Save method appends a comma and the Point's coordinates are appended separated by a comma. The Load method processes the added Point the same way the FromString method does. The Load method also sets the sprite's position based on the source tile.

For the CharacterLayer I added in a method that gets a character based on their name. Add the following method to the CharacterLayer class.

```

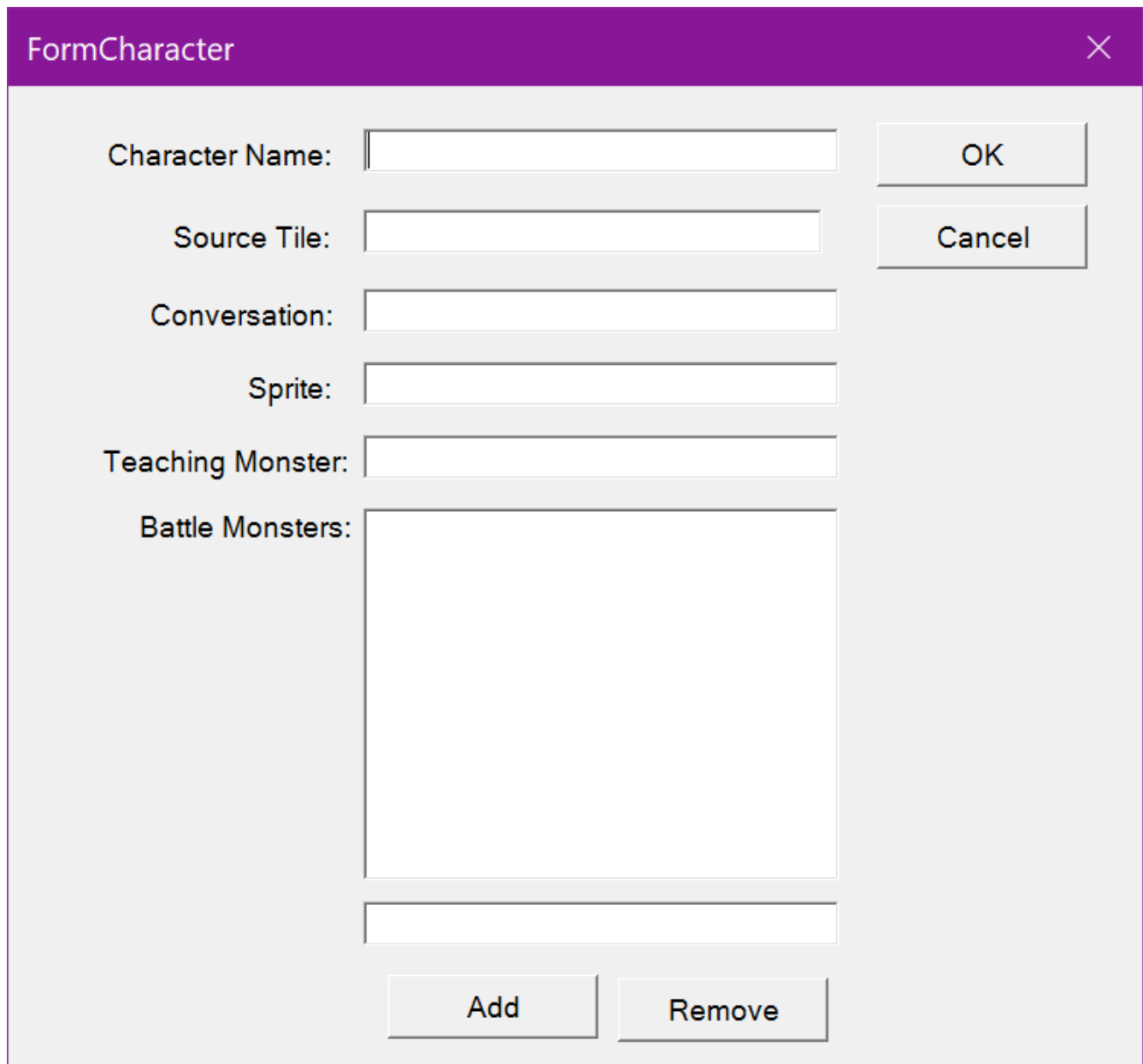
public Character GetCharacter(string v)
{
    foreach (Character c in characters.Values)
    {
        if (c.Name == v)
        {
            return c;
        }
    }

    return null;
}

```

What it does is loop over all of the characters in the characters collection. If the name of the character equals the value passed in the character is returned. If a character with that name is not found null is returned.

Next I added a form for creating a character. The finished for should look like this.



The screenshot shows a dialog box titled "FormCharacter" with a purple header bar and a close button (X) in the top right corner. The dialog contains several input fields and buttons:

- Character Name:** A text input field.
- Source Tile:** A text input field.
- Conversation:** A text input field.
- Sprite:** A text input field.
- Teaching Monster:** A text input field.
- Battle Monsters:** A large text area for listing monsters.
- Buttons:** "OK" and "Cancel" buttons are on the right side. "Add" and "Remove" buttons are at the bottom, below a small text input field.

The form works by filling out the fields. For the Source Tile field You enter the coordinates separated by a colon like so 4:4. For the battle monsters You enter the name of the monster in the text box then click the Add button. To remove a monster select it in the list box and click the Remove button.

Right click the ShadowEditor project in the Solution Explore, select Add and then Form

(Windows Forms). Name this new form CharacterForm. Open the CharacterForm.Designer.cs file by expanding the node for it in the Solution Explorer. Replace the contents with the following code.

```
namespace ShadowEditor
{
    partial class CharacterForm
    {
        /// <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.BtnOK = new System.Windows.Forms.Button();
            this.BtnCancel = new System.Windows.Forms.Button();
            this.label1 = new System.Windows.Forms.Label();
            this.label2 = new System.Windows.Forms.Label();
            this.TxtName = new System.Windows.Forms.TextBox();
            this.TxtConversation = new System.Windows.Forms.TextBox();
            this.label3 = new System.Windows.Forms.Label();
            this.TxtSprite = new System.Windows.Forms.TextBox();
            this.label4 = new System.Windows.Forms.Label();
            this.TxtTeach = new System.Windows.Forms.TextBox();
            this.label5 = new System.Windows.Forms.Label();
            this.LBShadowMonsters = new System.Windows.Forms.ListBox();
            this.BtnAdd = new System.Windows.Forms.Button();
            this.BtnRemove = new System.Windows.Forms.Button();
            this.TxtShadowMonster = new System.Windows.Forms.TextBox();
            this.label6 = new System.Windows.Forms.Label();
            this.TxtSourceTile = new System.Windows.Forms.TextBox();
            this.SuspendLayout();
            //
            // 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(311, 13);
```

```

this.BtnOK.Name = "BtnOK";
this.BtnOK.Size = new System.Drawing.Size(75, 23);
this.BtnOK.TabIndex = 15;
this.BtnOK.Text = "OK";
this.BtnOK.UseVisualStyleBackColor = true;
this.BtnOK.Click += new System.EventHandler(this.BtnOK_Click);
//
// 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(311, 42);
this.BtnCancel.Name = "BtnCancel";
this.BtnCancel.Size = new System.Drawing.Size(75, 23);
this.BtnCancel.TabIndex = 16;
this.BtnCancel.Text = "Cancel";
this.BtnCancel.UseVisualStyleBackColor = true;
this.BtnCancel.Click += new System.EventHandler(this.BtnCancel_Click);
//
// label1
//
this.label1.AutoSize = true;
this.label1.Location = new System.Drawing.Point(34, 18);
this.label1.Name = "label1";
this.label1.Size = new System.Drawing.Size(87, 13);
this.label1.TabIndex = 0;
this.label1.Text = "Character Name:";
//
// label2
//
this.label2.AutoSize = true;
this.label2.Location = new System.Drawing.Point(49, 75);
this.label2.Name = "label2";
this.label2.Size = new System.Drawing.Size(72, 13);
this.label2.TabIndex = 4;
this.label2.Text = "Conversation:";
//
// TxtName
//
this.TxtName.Location = new System.Drawing.Point(127, 15);
this.TxtName.Name = "TxtName";
this.TxtName.Size = new System.Drawing.Size(172, 20);
this.TxtName.TabIndex = 1;
//
// TxtConversation
//
this.TxtConversation.Location = new System.Drawing.Point(127, 72);
this.TxtConversation.Name = "TxtConversation";
this.TxtConversation.Size = new System.Drawing.Size(172, 20);
this.TxtConversation.TabIndex = 5;
//
// label3
//
this.label3.AutoSize = true;
this.label3.Location = new System.Drawing.Point(84, 101);
this.label3.Name = "label3";
this.label3.Size = new System.Drawing.Size(37, 13);
this.label3.TabIndex = 6;
this.label3.Text = "Sprite:";
//

```

```

// TxtSprite
//
this.TxtSprite.Location = new System.Drawing.Point(127, 98);
this.TxtSprite.Name = "TxtSprite";
this.TxtSprite.Size = new System.Drawing.Size(172, 20);
this.TxtSprite.TabIndex = 7;
//
// label4
//
this.label4.AutoSize = true;
this.label4.Location = new System.Drawing.Point(32, 127);
this.label4.Name = "label4";
this.label4.Size = new System.Drawing.Size(89, 13);
this.label4.TabIndex = 8;
this.label4.Text = "Teaching ShadowMonster:";
//
// TxtTeach
//
this.TxtTeach.Location = new System.Drawing.Point(127, 124);
this.TxtTeach.Name = "TxtTeach";
this.TxtTeach.Size = new System.Drawing.Size(172, 20);
this.TxtTeach.TabIndex = 9;
//
// label5
//
this.label5.AutoSize = true;
this.label5.Location = new System.Drawing.Point(45, 150);
this.label5.Name = "label5";
this.label5.Size = new System.Drawing.Size(76, 13);
this.label5.TabIndex = 11;
this.label5.Text = "Battle ShadowMonsters:";
//
// LBShadowMonsters
//
this.LBShadowMonsters.FormattingEnabled = true;
this.LBShadowMonsters.Location = new System.Drawing.Point(127, 150);
this.LBShadowMonsters.Name = "LBShadowMonsters";
this.LBShadowMonsters.Size = new System.Drawing.Size(172, 134);
this.LBShadowMonsters.TabIndex = 11;
//
// BtnAdd
//
this.BtnAdd.Location = new System.Drawing.Point(136, 316);
this.BtnAdd.Name = "BtnAdd";
this.BtnAdd.Size = new System.Drawing.Size(75, 23);
this.BtnAdd.TabIndex = 13;
this.BtnAdd.Text = "Add";
this.BtnAdd.UseVisualStyleBackColor = true;
//
// BtnRemove
//
this.BtnRemove.Location = new System.Drawing.Point(218, 317);
this.BtnRemove.Name = "BtnRemove";
this.BtnRemove.Size = new System.Drawing.Size(75, 23);
this.BtnRemove.TabIndex = 14;
this.BtnRemove.Text = "Remove";
this.BtnRemove.UseVisualStyleBackColor = true;
//
// TxtShadowMonster
//

```

```

this.TxtShadowMonster.Location = new System.Drawing.Point(127, 290);
this.TxtShadowMonster.Name = "TxtShadowMonster";
this.TxtShadowMonster.Size = new System.Drawing.Size(172, 20);
this.TxtShadowMonster.TabIndex = 12;
//
// label6
//
this.label6.AutoSize = true;
this.label6.Location = new System.Drawing.Point(57, 47);
this.label6.Name = "label6";
this.label6.Size = new System.Drawing.Size(64, 13);
this.label6.TabIndex = 2;
this.label6.Text = "Source Tile:";
//
// TxtSourceTile
//
this.TxtSourceTile.Location = new System.Drawing.Point(127, 44);
this.TxtSourceTile.Name = "TxtSourceTile";
this.TxtSourceTile.Size = new System.Drawing.Size(166, 20);
this.TxtSourceTile.TabIndex = 3;
//
// FormCharacter
//
this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
this.ClientSize = new System.Drawing.Size(406, 519);
this.Controls.Add(this.TxtSourceTile);
this.Controls.Add(this.label6);
this.Controls.Add(this.TxtShadowMonster);
this.Controls.Add(this.BtnRemove);
this.Controls.Add(this.BtnAdd);
this.Controls.Add(this.LBShadowMonsters);
this.Controls.Add(this.label5);
this.Controls.Add(this.TxtTeach);
this.Controls.Add(this.label4);
this.Controls.Add(this.TxtSprite);
this.Controls.Add(this.label3);
this.Controls.Add(this.TxtConversation);
this.Controls.Add(this.TxtName);
this.Controls.Add(this.label2);
this.Controls.Add(this.label1);
this.Controls.Add(this.BtnCancel);
this.Controls.Add(this.BtnOK);
this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedDialog;
this.MaximizeBox = false;
this.MinimizeBox = false;
this.Name = "FormCharacter";
this.Text = "FormCharacter";
this.Load += new System.EventHandler(this.FormCharacter_Load);
this.ResumeLaYout(false);
this.PerformLaYout();
}

#endregion

private System.Windows.Forms.Button BtnOK;
private System.Windows.Forms.Button BtnCancel;
private System.Windows.Forms.Label label1;
private System.Windows.Forms.Label label2;

```

```

        internal System.Windows.Forms.TextBox TxtName;
        private System.Windows.Forms.TextBox TxtConversation;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.TextBox TxtSprite;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.TextBox TxtTeach;
        private System.Windows.Forms.Label label5;
        private System.Windows.Forms.ListBox LBShadowMonsters;
        private System.Windows.Forms.Button BtnAdd;
        private System.Windows.Forms.Button BtnRemove;
        private System.Windows.Forms.TextBox TxtShadowMonster;
        private System.Windows.Forms.Label label6;
        private System.Windows.Forms.TextBox TxtSourceTile;
    }
}

```

The code is generated by Visual Studio so I'm not going to explain it. Now select the CharacterForm in the Solution Explorer and press F7 to open the code for the Character form or right click CharacterForm and select View Code. Replace the code with the following.

```

using ShadowMonsters.Characters;
using ShadowMonsters.TileEngine;
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 ShadowEditor
{
    public partial class CharacterForm : Form
    {
        public string Character;
        private readonly Character c;

        public bool OKPressed { get; private set; }

        public CharacterForm()
        {
            InitializeComponent();
        }

        public CharacterForm(Character c)
        {
            InitializeComponent();
            this.c = c;
        }

        private void FormCharacter_Load(object sender, EventArgs e)
        {
            BtnAdd.Click += BtnAdd_Click;
            BtnRemove.Click += BtnRemove_Click;

            if (c != null)
            {

```

```

        TxtName.Text = c.Name;
        TxtConversation.Text = c.Conversation;
        TxtSprite.Text = c.SpriteName;
        if (c.GiveMonster != null)
        {
            TxtTeach.Text = c.GiveMonster.Name;
        }

        Point source = new Point(
            (int)c.Sprite.Position.X / Engine.TileWidth,
            (int)c.Sprite.Position.Y / Engine.TileHeight);

        TxtSourceTile.Text = source.X + ":" + source.Y + ",";

        foreach (var a in c.BattleMonsters)
        {
            if (a != null)
            {
                LBShadowMonsters.Items.Add(a.Name);
            }
        }
    }
}

private void BtnRemove_Click(object sender, EventArgs e)
{
    if (LBShadowMonsters.SelectedIndex >= 0)
    {
        LBShadowMonsters.Items.RemoveAt(LBShadowMonsters.SelectedIndex);
    }
}

private void BtnAdd_Click(object sender, EventArgs e)
{
    if (LBShadowMonsters.Items.Count <= 6 && !
string.IsNullOrEmpty(TxtShadowMonster.Text))
    {
        LBShadowMonsters.Items.Add(TxtShadowMonster.Text);
    }
}

private void BtnCancel_Click(object sender, EventArgs e)
{
    OKPressed = false;
    Close();
}

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

    if (string.IsNullOrEmpty(TxtSourceTile.Text))
    {
        MessageBox.Show("You must enter the source tile for the character");
        return;
    }
}

```

```

        if (string.IsNullOrEmpty(TxtSprite.Text))
        {
            MessageBox.Show("You must enter the name of the sprite for the character.");
            return;
        }

        if (string.IsNullOrEmpty(TxtConversation.Text))
        {
            MessageBox.Show("You must enter the name of the conversation.");
            return;
        }

        Character = TxtName.Text + "," +
            TxtSprite.Text + "," +
            "WalkDown," +
            TxtConversation.Text + ",0," +
            TxtSourceTile.Text + ",";

        foreach (var v in LBShadowMonsters.Items)
        {
            Character += v.ToString() + ",";
        }

        for (int i = LBShadowMonsters.Items.Count; i < 6; i++)
            Character += ",";

        Character += TxtTeach.Text;

        OKPressed = true;
        Close();
    }

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

```

There is a public string field that will return the character as a string that can be processed by the editor. There is a read only field that holds a Character that is passed to the form in the constructor. There is a property OKPressed that tells the calling form how the form was closed. There is a parameterless constructor that calls InitializeComponent. There is also a constructor that takes a Character parameter that calls InitializeComponent and sets the c field to the value passed in. In the form Load event handler I wire event handlers for the buttons to add and remove shadow monsters. I then check to see if the c field is not null. If it is not null I populate the controls of the form using the c field.

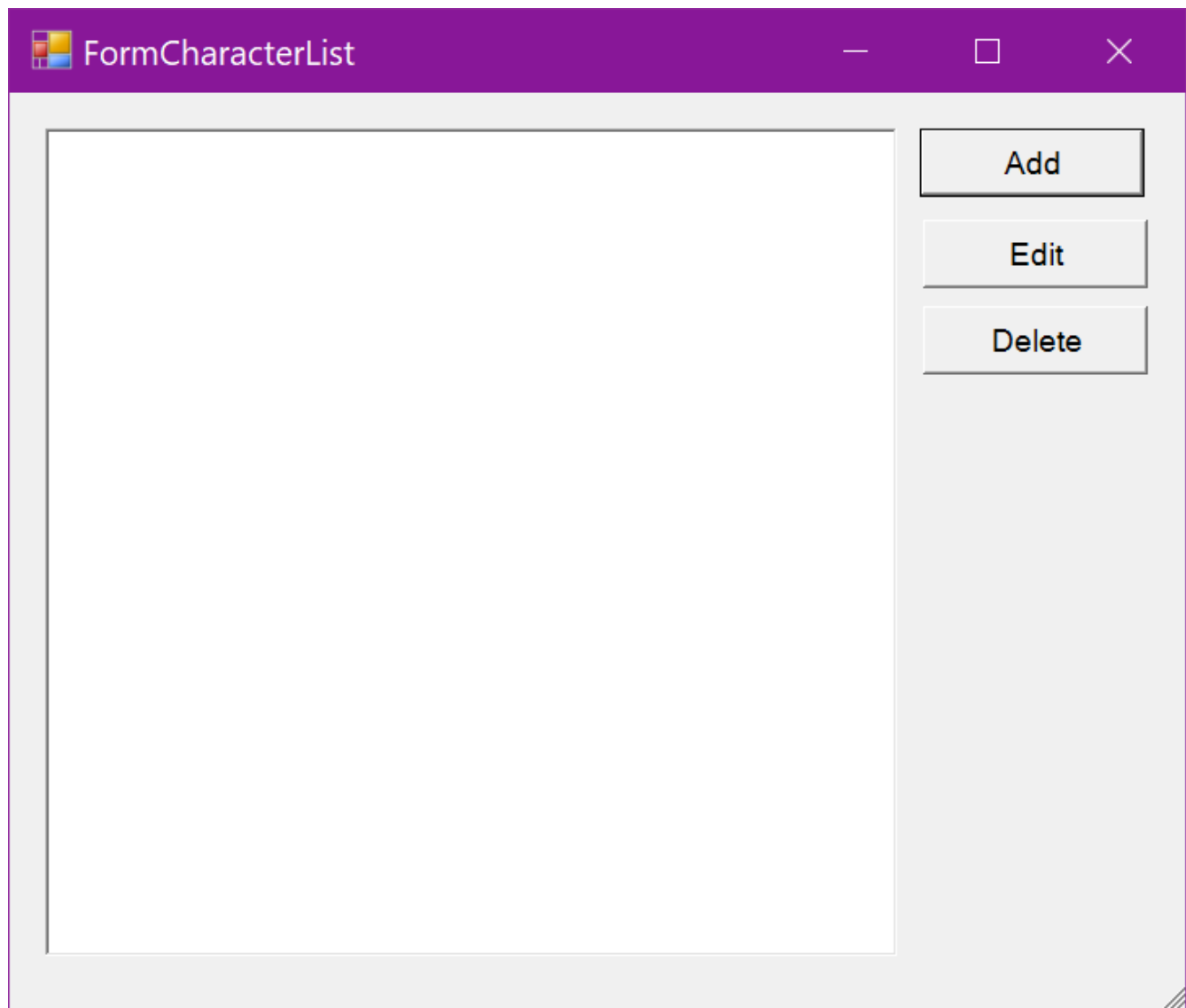
The event handler for the remove button checks to see if the SelectedIndex property of LBShadowMonsters is greater than or equal to zero. If it is I remove the selected index.

The event handler for the add button checks to see if there are less than or equal to 6 items in LBShadowMonsters and that there is text in the TxtShadowMonster text box. If both of those conditions are true I add the text in TxtShadowMonster to the items in LBShadowMonsters.

The event handler for the Cancel button I set OKPressed to be false and then close the form.

The event handler for the OK button validates that there is text in the TxtName, TxtSourceTile, TxtSprite and TxtConversation. I then create the Character string for the character. OKPressed is set to true and the form is closed.

Now I'm going to add a form for adding characters to the map. It looks like this.



The form works by using the buttons on the right. The Add button will add a new character. The Edit button will edit the character in the list box. The Delete button will delete the character selected in the list box.

Right click the ShadowEditor project, select Add and then Form (Windows Forms). Name this new form CharacterListForm. Open the CharacterListForm.Designer.cs file with the following code.

```
namespace ShadowEditor
{
    partial class CharacterListForm
```



```

{
    /// <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.LBCharacters = new System.Windows.Forms.ListBox();
        this.BtnAdd = new System.Windows.Forms.Button();
        this.BtnEdit = new System.Windows.Forms.Button();
        this.BtnDelete = new System.Windows.Forms.Button();
        this.SuspendLayout();
        //
        // LBCharacters
        //
        this.LBCharacters.FormatEnabled = true;
        this.LBCharacters.Location = new System.Drawing.Point(12, 12);
        this.LBCharacters.Name = "LBCharacters";
        this.LBCharacters.Size = new System.Drawing.Size(287, 277);
        this.LBCharacters.TabIndex = 0;
        //
        // BtnAdd
        //
        this.BtnAdd.Location = new System.Drawing.Point(305, 12);
        this.BtnAdd.Name = "BtnAdd";
        this.BtnAdd.Size = new System.Drawing.Size(75, 23);
        this.BtnAdd.TabIndex = 1;
        this.BtnAdd.Text = "Add";
        this.BtnAdd.UseVisualStyleBackColor = true;
        //
        // BtnEdit
        //
        this.BtnEdit.Location = new System.Drawing.Point(306, 42);
        this.BtnEdit.Name = "BtnEdit";
        this.BtnEdit.Size = new System.Drawing.Size(75, 23);
        this.BtnEdit.TabIndex = 2;
        this.BtnEdit.Text = "Edit";
        this.BtnEdit.UseVisualStyleBackColor = true;
        //

```

```

        // BtnDelete
        //
        this.BtnDelete.Location = new System.Drawing.Point(306, 71);
        this.BtnDelete.Name = "BtnDelete";
        this.BtnDelete.Size = new System.Drawing.Size(75, 23);
        this.BtnDelete.TabIndex = 3;
        this.BtnDelete.Text = "Delete";
        this.BtnDelete.UseVisualStyleBackColor = true;
        //
        // FormCharacterList
        //
        this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
        this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
        this.ClientSize = new System.Drawing.Size(394, 305);
        this.Controls.Add(this.BtnDelete);
        this.Controls.Add(this.BtnEdit);
        this.Controls.Add(this.BtnAdd);
        this.Controls.Add(this.LBCharacters);
        this.Name = "FormCharacterList";
        this.Text = "FormCharacterList";
        this.ResumeLayout(false);
    }

    #endregion

    private System.Windows.Forms.ListBox LBCharacters;
    private System.Windows.Forms.Button BtnAdd;
    private System.Windows.Forms.Button BtnEdit;
    private System.Windows.Forms.Button BtnDelete;
}

```

This is all generated code so I'm not going to explain it all. What I am going to explain is the way the form works. There is a list box that holds all of the characters on the map. There are then buttons to add, edit and delete characters.

Now I'm going to add the logic to the form. Select the form in the Solution Explorer and press the F7 key or right click the form in the Solution Explorer and select View Code. Replace the code with the following.

```

using ShadowMonsters.Characters;
using ShadowMonsters.TileEngine;
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;
using Microsoft.Xna.Framework;

namespace ShadowEditor
{
    public partial class CharacterListForm : Form
    {

```

```

private readonly TileMap map;
private readonly Editor game;

public CharacterListForm(TileMap map, Editor game)
{
    InitializeComponent();

    this.map = map;
    this.game = game;

    foreach (var c in map.CharacterLayer.Characters.Keys)
    {
        if (!(map.CharacterLayer.Characters[c] is Merchant))
        {
            if (map.CharacterLayer.Characters[c].Name != null)
            {
                LBCharacters.Items.Add(map.CharacterLayer.Characters[c].Name);
            }
            else
            {
                LBCharacters.Items.Add("Unknown");
            }
        }
    }

    BtnAdd.Click += BtnAdd_Click;
    BtnEdit.Click += BtnEdit_Click;
    BtnDelete.Click += BtnDelete_Click;
}

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

    if (character.OKPressed)
    {
        Character c = Character.FromString(game, character.Character);

        map.CharacterLayer.Characters.Add(
            c.SourceTile,
            c);
        LBCharacters.Items.Add(c.Name);
    }
}

private void BtnEdit_Click(object sender, EventArgs e)
{
    if (LBCharacters.SelectedIndex < 0 || LBCharacters.Items.Count == 0)
    {
        return;
    }

    if (LBCharacters.SelectedItem.ToString() == "Unknown")
    {
        return;
    }

    Character c =
map.CharacterLayer.GetCharacter(LBCharacters.SelectedItem.ToString());

```

```

        CharacterForm formCharacter = new CharacterForm(c);

        formCharacter.TxtName.Enabled = false;
        formCharacter.ShowDialog();

        if (formCharacter.OKPressed)
        {
            map.CharacterLayer.Characters[c.SourceTile] =
                Character.FromString(game, formCharacter.Character);
        }
    }

    private void BtnDelete_Click(object sender, EventArgs e)
    {
        if (LBCharacters.SelectedIndex < 0)
        {
            return;
        }

        map.CharacterLayer.Characters.Remove(
            map.CharacterLayer.GetCharacter(LBCharacters.SelectedItem.ToString()).SourceTile);
        LBCharacters.Items.RemoveAt(LBCharacters.SelectedIndex);
    }
}

```

There is a TileMap field and an Editor field in this class. The TileMap field gives us access to the CharacterLayer. The Editor is for the FromString method of the Character class. The constructor requires a TileMap parameter and an Editor parameter. It sets the fields to the parameters that are passed in. It then loops over all of the Keys in the Characters collection. It checks to see if the character is not a Merchant. If its Name property is not null the name is added to the Items collection of the list box that holds the characters. If it is null Unknown is added. It then wires the event handlers for the Add, Edit and Delete buttons.

In the Click event handler for BtnAdd I create a CharacterForm then call the ShowDialog method. If the OKPressed property of the form is true I create a Character using the FromString method. I then add the character to the character layer using the SourceTile of the character for the key. I then add the name to the list of characters.

In the event handler for the Click event of the Edit button I check to see if there is a selected item or the number of items equals zero. If either are true I exit the method. If the selected character is Unknown I exit the method. I then use the GetCharacter method of the character layer to get the character with the selected name. I then create a CharacterForm and pass in the character so the form will be populated. I set TxtName's Enabled property to false because I don't want it edited because it is a key field. Since the SourceTile is also a key that should probably be disabled as well. I then call the ShowDialog method to display the form. If the OK button was pressed I set the character to the updated character.

In the event handler for the Click event of the Delete button I check to see if the SelectedIndex property is less than 0 and if it is I exit the method. I then call the Remove method of the Characters collection of the character layer passing in the SourceTile property of the character with the selected name. I then remove the entry from the list box holding the

character names.

Before getting to creating merchants I need to update the FromString and Load methods of the Merchant class to handle the source file. They are the same changes that were made to the Character class so I'm not going to explain it. Replace the FromString and Load methods of the Merchant class with the following.

```
public new static Merchant FromString(Game game, string characterString)
{
    if (gameRef == null)
    {
        gameRef = game;
    }

    Merchant character = new Merchant();
    character.backpack = new Backpack();
    string[] parts = characterString.Split(',');

    character.name = parts[0];
    character.textureName = parts[1];
    character.sprite = new AnimatedSprite(
        game.Content.Load<Texture2D>(@"CharacterSprites\" + parts[1]),
        Game1.Animations)
    {
        CurrentAnimation = (AnimationKey)Enum.Parse(typeof(AnimationKey), parts[2])
    };
    character.conversation = parts[3];
    character.currentMonster = int.Parse(parts[4]);
    string[] items = parts[5].Split(':');
    character.SourceTile = new Point(int.Parse(items[0]), int.Parse(items[1]));
    character.sprite.Position = new Vector2(
        character.SourceTile.X * Engine.TileWidth,
        character.SourceTile.Y * Engine.TileHeight);
    for (int i = 6; i < 12 && i < parts.Length - 1; i++)
    {
        ShadowMonster monster =
ShadowMonsterManager.GetShadowMonster(parts[i].ToLowerInvariant());
        character.monsters[i - 6] = monster;
    }

    character.givingMonster = ShadowMonsterManager.GetShadowMonster(parts[parts.Length
- 1].ToLowerInvariant());
    return character;
}

new public static Merchant Load(ContentManager content, BinaryReader reader)
{
    Merchant c = new Merchant
    {
        backpack = new Backpack()
    };

    string data = reader.ReadString();
    string[] parts = data.Split(',');
    reader.ReadInt32();

    c.name = parts[0];
    c.textureName = parts[1];
    c.sprite = new AnimatedSprite(
```

```

        content.Load<Texture2D>(
            @"CharacterSprites\" + parts[1]),
        Game1.Animations);

parts[2]);
c.sprite.CurrentAnimation = (AnimationKey)Enum.Parse(typeof(AnimationKey),
c.conversation = parts[3];
c.currentMonster = int.Parse(parts[4]);
c.Battled = bool.Parse(parts[5]);
string[] items = parts[6].Split(':');
c.SourceTile = new Point(int.Parse(items[0]), int.Parse(items[1]));
c.sprite.Position = new Vector2(
    c.SourceTile.X * Engine.TileWidth,
    c.SourceTile.Y * Engine.TileHeight);

for (int i = 0; i < 6; i++)
{
    string avatar = reader.ReadString();

    if (avatar != "")
    {
        c.monsters[i] = ShadowMonster.Load(content, avatar);
    }

    reader.ReadInt32();
}

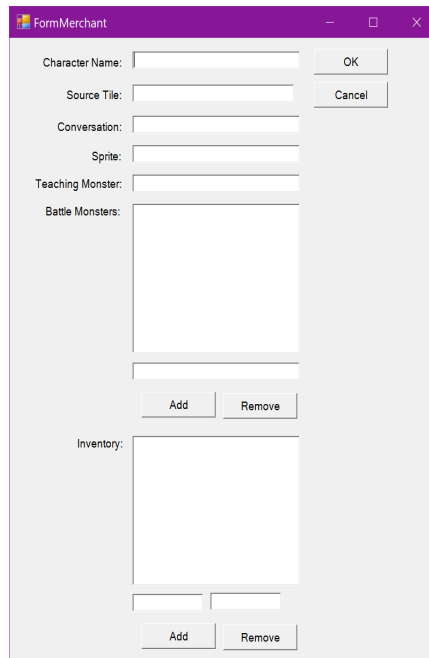
string giving = reader.ReadString();

if (giving != "")
{
    c.givingMonster = ShadowMonster.Load(content, giving);
}

c.backpack = Backpack.Load(reader);
return c;
}

```

I'm now going to add a form for creating merchants. The finished form looks like the following.



Right click the ShadowEditor project in the Solution Explorer, select Add and then Form (Windows Forms). Name this new form MerchantForm. Replace the contents of the MerchantForm.Designer.cs to the following.

```
namespace ShadowEditor
{
    partial class MerchantForm
    {
        /// <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.TxtSourceTile = new System.Windows.Forms.TextBox();
```

```

this.label6 = new System.Windows.Forms.Label();
this.TxtShadowMonster = new System.Windows.Forms.TextBox();
this.BtnRemove = new System.Windows.Forms.Button();
this.BtnAdd = new System.Windows.Forms.Button();
this.LBShadowMonsters = new System.Windows.Forms.ListBox();
this.label5 = new System.Windows.Forms.Label();
this.TxtTeach = new System.Windows.Forms.TextBox();
this.label4 = new System.Windows.Forms.Label();
this.TxtSprite = new System.Windows.Forms.TextBox();
this.label3 = new System.Windows.Forms.Label();
this.TxtConversation = new System.Windows.Forms.TextBox();
this.TxtName = new System.Windows.Forms.TextBox();
this.label2 = new System.Windows.Forms.Label();
this.label1 = new System.Windows.Forms.Label();
this.BtnCancel = new System.Windows.Forms.Button();
this.BtnOK = new System.Windows.Forms.Button();
this.label7 = new System.Windows.Forms.Label();
this.lbBackpack = new System.Windows.Forms.ListBox();
this.BtnAddItem = new System.Windows.Forms.Button();
this.BtnRemoveItem = new System.Windows.Forms.Button();
this.TxtItem = new System.Windows.Forms.TextBox();
this.TxtCount = new System.Windows.Forms.TextBox();
this.SuspendLayout();
//
// TxtSourceTile
//
this.TxtSourceTile.Location = new System.Drawing.Point(229, 76);
this.TxtSourceTile.Margin = new System.Windows.Forms.Padding(6);
this.TxtSourceTile.Name = "TxtSourceTile";
this.TxtSourceTile.Size = new System.Drawing.Size(301, 29);
this.TxtSourceTile.TabIndex = 20;
//
// label6
//
this.label6.AutoSize = true;
this.label6.Location = new System.Drawing.Point(103, 81);
this.label6.Margin = new System.Windows.Forms.Padding(6, 0, 6, 0);
this.label6.Name = "label6";
this.label6.Size = new System.Drawing.Size(118, 25);
this.label6.TabIndex = 19;
this.label6.Text = "Source Tile:";
//
// TxtShadowMonster
//
this.TxtShadowMonster.Location = new System.Drawing.Point(229, 530);
this.TxtShadowMonster.Margin = new System.Windows.Forms.Padding(6);
this.TxtShadowMonster.Name = "TxtShadowMonster";
this.TxtShadowMonster.Size = new System.Drawing.Size(312, 29);
this.TxtShadowMonster.TabIndex = 29;
//
// BtnRemove
//
this.BtnRemove.Location = new System.Drawing.Point(398, 580);
this.BtnRemove.Margin = new System.Windows.Forms.Padding(6);
this.BtnRemove.Name = "BtnRemove";
this.BtnRemove.Size = new System.Drawing.Size(138, 42);
this.BtnRemove.TabIndex = 31;
this.BtnRemove.Text = "Remove";
this.BtnRemove.UseVisualStyleBackColor = true;
//

```



```

// BtnAdd
//
this.BtnAdd.Location = new System.Drawing.Point(246, 578);
this.BtnAdd.Margin = new System.Windows.Forms.Padding(6);
this.BtnAdd.Name = "BtnAdd";
this.BtnAdd.Size = new System.Drawing.Size(138, 42);
this.BtnAdd.TabIndex = 30;
this.BtnAdd.Text = "Add";
this.BtnAdd.UseVisualStyleBackColor = true;
//
// LBSHadowMonsters
//
this.LBSHadowMonsters.FormattingEnabled = true;
this.LBSHadowMonsters.ItemHeight = 24;
this.LBSHadowMonsters.Location = new System.Drawing.Point(229, 271);
this.LBSHadowMonsters.Margin = new System.Windows.Forms.Padding(6);
this.LBSHadowMonsters.Name = "LBSHadowMonsters";
this.LBSHadowMonsters.Size = new System.Drawing.Size(312, 244);
this.LBSHadowMonsters.TabIndex = 28;
//
// label5
//
this.label5.AutoSize = true;
this.label5.Location = new System.Drawing.Point(64, 271);
this.label5.Margin = new System.Windows.Forms.Padding(6, 0, 6, 0);
this.label5.Name = "label5";
this.label5.Size = new System.Drawing.Size(153, 25);
this.label5.TabIndex = 27;
this.label5.Text = "Battle Monsters:";
//
// TxtTeach
//
this.TxtTeach.Location = new System.Drawing.Point(229, 223);
this.TxtTeach.Margin = new System.Windows.Forms.Padding(6);
this.TxtTeach.Name = "TxtTeach";
this.TxtTeach.Size = new System.Drawing.Size(312, 29);
this.TxtTeach.TabIndex = 26;
//
// label4
//
this.label4.AutoSize = true;
this.label4.Location = new System.Drawing.Point(45, 226);
this.label4.Margin = new System.Windows.Forms.Padding(6, 0, 6, 0);
this.label4.Name = "label4";
this.label4.Size = new System.Drawing.Size(176, 25);
this.label4.TabIndex = 25;
this.label4.Text = "Teaching Monster:";
//
// TxtSprite
//
this.TxtSprite.Location = new System.Drawing.Point(229, 175);
this.TxtSprite.Margin = new System.Windows.Forms.Padding(6);
this.TxtSprite.Name = "TxtSprite";
this.TxtSprite.Size = new System.Drawing.Size(312, 29);
this.TxtSprite.TabIndex = 24;
//
// label3
//
this.label3.AutoSize = true;
this.label3.Location = new System.Drawing.Point(150, 181);

```

```

this.label3.Margin = new System.Windows.Forms.Padding(6, 0, 6, 0);
this.label3.Name = "label3";
this.label3.Size = new System.Drawing.Size(69, 25);
this.label3.TabIndex = 23;
this.label3.Text = "Sprite:";
//
// TxtConversation
//
this.TxtConversation.Location = new System.Drawing.Point(229, 127);
this.TxtConversation.Margin = new System.Windows.Forms.Padding(6);
this.TxtConversation.Name = "TxtConversation";
this.TxtConversation.Size = new System.Drawing.Size(312, 29);
this.TxtConversation.TabIndex = 22;
//
// TxtName
//
this.TxtName.Location = new System.Drawing.Point(229, 22);
this.TxtName.Margin = new System.Windows.Forms.Padding(6);
this.TxtName.Name = "TxtName";
this.TxtName.Size = new System.Drawing.Size(312, 29);
this.TxtName.TabIndex = 18;
//
// label2
//
this.label2.AutoSize = true;
this.label2.Location = new System.Drawing.Point(86, 133);
this.label2.Margin = new System.Windows.Forms.Padding(6, 0, 6, 0);
this.label2.Name = "label2";
this.label2.Size = new System.Drawing.Size(134, 25);
this.label2.TabIndex = 21;
this.label2.Text = "Conversation:";
//
// label1
//
this.label1.AutoSize = true;
this.label1.Location = new System.Drawing.Point(59, 28);
this.label1.Margin = new System.Windows.Forms.Padding(6, 0, 6, 0);
this.label1.Name = "label1";
this.label1.Size = new System.Drawing.Size(161, 25);
this.label1.TabIndex = 17;
this.label1.Text = "Character Name:";
//
// 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(567, 72);
this.BtnCancel.Margin = new System.Windows.Forms.Padding(6);
this.BtnCancel.Name = "BtnCancel";
this.BtnCancel.Size = new System.Drawing.Size(138, 42);
this.BtnCancel.TabIndex = 33;
this.BtnCancel.Text = "Cancel";
this.BtnCancel.UseVisualStyleBackColor = true;
this.BtnCancel.Click += new System.EventHandler(this.BtnCancel_Click);
//
// 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(567, 18);

```

```

this.BtnOK.Margin = new System.Windows.Forms.Padding(6);
this.BtnOK.Name = "BtnOK";
this.BtnOK.Size = new System.Drawing.Size(138, 42);
this.BtnOK.TabIndex = 32;
this.BtnOK.Text = "OK";
this.BtnOK.UseVisualStyleBackColor = true;
this.BtnOK.Click += new System.EventHandler(this.BtnOK_Click);
//
// label7
//
this.label7.AutoSize = true;
this.label7.Location = new System.Drawing.Point(123, 650);
this.label7.Margin = new System.Windows.Forms.Padding(6, 0, 6, 0);
this.label7.Name = "label7";
this.label7.Size = new System.Drawing.Size(98, 25);
this.label7.TabIndex = 27;
this.label7.Text = "Inventory:";
//
// lbBackpack
//
this.lbBackpack.FormattingEnabled = true;
this.lbBackpack.ItemHeight = 24;
this.lbBackpack.Location = new System.Drawing.Point(229, 650);
this.lbBackpack.Margin = new System.Windows.Forms.Padding(6);
this.lbBackpack.Name = "lbBackpack";
this.lbBackpack.Size = new System.Drawing.Size(312, 244);
this.lbBackpack.TabIndex = 28;
this.lbBackpack.SelectedIndexChanged += new
System.EventHandler(this.lbBackpack_SelectedIndexChanged);
//
// BtnAddItem
//
this.BtnAddItem.Location = new System.Drawing.Point(246, 956);
this.BtnAddItem.Margin = new System.Windows.Forms.Padding(6);
this.BtnAddItem.Name = "BtnAddItem";
this.BtnAddItem.Size = new System.Drawing.Size(138, 42);
this.BtnAddItem.TabIndex = 30;
this.BtnAddItem.Text = "Add";
this.BtnAddItem.UseVisualStyleBackColor = true;
this.BtnAddItem.Click += new System.EventHandler(this.BtnAddItem_Click);
//
// BtnRemoveItem
//
this.BtnRemoveItem.Location = new System.Drawing.Point(398, 958);
this.BtnRemoveItem.Margin = new System.Windows.Forms.Padding(6);
this.BtnRemoveItem.Name = "BtnRemoveItem";
this.BtnRemoveItem.Size = new System.Drawing.Size(138, 42);
this.BtnRemoveItem.TabIndex = 31;
this.BtnRemoveItem.Text = "Remove";
this.BtnRemoveItem.UseVisualStyleBackColor = true;
this.BtnRemoveItem.Click += new System.EventHandler(this.BtnRemoveItem_Click);
//
// TxtItem
//
this.TxtItem.Location = new System.Drawing.Point(229, 908);
this.TxtItem.Margin = new System.Windows.Forms.Padding(6);
this.TxtItem.Name = "TxtItem";
this.TxtItem.Size = new System.Drawing.Size(132, 29);
this.TxtItem.TabIndex = 29;
//

```

```

        // TxtCount
        //
        this.TxtCount.Location = new System.Drawing.Point(374, 906);
        this.TxtCount.Margin = new System.Windows.Forms.Padding(6);
        this.TxtCount.Name = "TxtCount";
        this.TxtCount.Size = new System.Drawing.Size(132, 29);
        this.TxtCount.TabIndex = 29;
        //
        // MerchantForm
        //
        this.AutoScaleDimensions = new System.Drawing.SizeF(11F, 24F);
        this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
        this.ClientSize = new System.Drawing.Size(799, 1015);
        this.Controls.Add(this.TxtSourceTile);
        this.Controls.Add(this.label6);
        this.Controls.Add(this.TxtCount);
        this.Controls.Add(this.TxtItem);
        this.Controls.Add(this.TxtShadowMonster);
        this.Controls.Add(this.BtnRemoveItem);
        this.Controls.Add(this.BtnAddItem);
        this.Controls.Add(this.BtnRemove);
        this.Controls.Add(this.lbBackpack);
        this.Controls.Add(this.BtnAdd);
        this.Controls.Add(this.label7);
        this.Controls.Add(this.LBShadowMonsters);
        this.Controls.Add(this.label5);
        this.Controls.Add(this.TxtTeach);
        this.Controls.Add(this.label4);
        this.Controls.Add(this.TxtSprite);
        this.Controls.Add(this.label3);
        this.Controls.Add(this.TxtConversation);
        this.Controls.Add(this.TxtName);
        this.Controls.Add(this.label2);
        this.Controls.Add(this.label1);
        this.Controls.Add(this.BtnCancel);
        this.Controls.Add(this.BtnOK);
        this.Margin = new System.Windows.Forms.Padding(4);
        this.Name = "MerchantForm";
        this.Text = "FormMerchant";
        this.Load += new System.EventHandler(this.FormMerchant_Load);
        this.ResumeLayout(false);
        this.PerformLayout();
    }

```

```

}

```

```

#endregion

```

```

private System.Windows.Forms.TextBox TxtSourceTile;
private System.Windows.Forms.Label label6;
private System.Windows.Forms.TextBox TxtShadowMonster;
private System.Windows.Forms.Button BtnRemove;
private System.Windows.Forms.Button BtnAdd;
private System.Windows.Forms.ListBox LBShadowMonsters;
private System.Windows.Forms.Label label5;
private System.Windows.Forms.TextBox TxtTeach;
private System.Windows.Forms.Label label4;
private System.Windows.Forms.TextBox TxtSprite;
private System.Windows.Forms.Label label3;
private System.Windows.Forms.TextBox TxtConversation;
internal System.Windows.Forms.TextBox TxtName;

```

```

        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Button BtnCancel;
        private System.Windows.Forms.Button BtnOK;
        private System.Windows.Forms.Label label7;
        private System.Windows.Forms.ListBox lbBackpack;
        private System.Windows.Forms.Button BtnAddItem;
        private System.Windows.Forms.Button BtnRemoveItem;
        private System.Windows.Forms.TextBox TxtItem;
        private System.Windows.Forms.TextBox TxtCount;
    }
}

```

Now I'm going to add the logic for the form. Select the MerchantForm in the Solution Explorer and press F7 or right click the MerchantForm and select View Code. Replace the code with the following.

```

using ShadowMonsters.Characters;
using ShadowMonsters.TileEngine;
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 ShadowEditor
{
    public partial class MerchantForm : Form
    {
        public bool OKPressed { get; internal set; }
        public string Character { get; internal set; }
        public string Backpack { get; internal set; }

        public MerchantForm()
        {
            InitializeComponent();
        }

        public MerchantForm(Merchant c)
        {
            InitializeComponent();

            if (c != null)
            {
                TxtName.Text = c.Name;
                TxtConversation.Text = c.Conversation;
                TxtSprite.Text = c.SpriteName;
                if (c.GiveMonster != null)
                {
                    TxtTeach.Text = c.GiveMonster.Name;
                }

                Point source = new Point(
                    (int)c.Sprite.Position.X / Engine.TileWidth,
                    (int)c.Sprite.Position.Y / Engine.TileHeight);
            }
        }
    }
}

```

```

        TxtSourceTile.Text = source.X + ":" + source.Y;

        foreach (var a in c.BattleMonsters)
        {
            if (a != null)
            {
                LBShadowMonsters.Items.Add(a.Name);
            }
        }

        foreach (var i in c.Backpack.Items)
        {
            lbBackpack.Items.Add(i.Name + ":" + i.Count);
        }
    }

    private void FormMerchant_Load(object sender, EventArgs e)
    {
        BtnAdd.Click += BtnAdd_Click;
        BtnRemove.Click += BtnRemove_Click;
    }

    private void BtnRemove_Click(object sender, EventArgs e)
    {
        if (LBShadowMonsters.SelectedIndex >= 0)
        {
            LBShadowMonsters.Items.RemoveAt(LBShadowMonsters.SelectedIndex);
        }
    }

    private void BtnAdd_Click(object sender, EventArgs e)
    {
        if (LBShadowMonsters.Items.Count <= 6 && !
string.IsNullOrEmpty(TxtShadowMonster.Text))
        {
            LBShadowMonsters.Items.Add(TxtShadowMonster.Text);
        }
    }

    private void BtnAddItem_Click(object sender, EventArgs e)
    {
        lbBackpack.Items.Add(TxtItem.Text + ":" + TxtCount.Text);
    }

    private void BtnRemoveItem_Click(object sender, EventArgs e)
    {
        if (lbBackpack.SelectedIndex >= 0)
        {
            lbBackpack.Items.RemoveAt(lbBackpack.SelectedIndex);
        }
    }

    private void BtnCancel_Click(object sender, EventArgs e)
    {
        OKPressed = false;
        Close();
    }

```

```

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

    if (string.IsNullOrEmpty(TxtSourceTile.Text))
    {
        MessageBox.Show("You must enter the source tile for the character");
        return;
    }
    if (string.IsNullOrEmpty(TxtSprite.Text))
    {
        MessageBox.Show("You must enter the name of the sprite for the character.");
        return;
    }

    if (string.IsNullOrEmpty(TxtConversation.Text))
    {
        MessageBox.Show("You mut enter the name of the conversation.");
        return;
    }

    Character = TxtName.Text + "," +
        TxtSprite.Text + "," +
        "WalkDown," +
        TxtConversation.Text + ",0," +
        TxtSourceTile.Text + ",";

    foreach (var v in LBShadowMonsters.Items)
    {
        Character += v.ToString() + ",";
    }

    for (int i = LBShadowMonsters.Items.Count; i < 6; i++)
        Character += ",";

    Character += TxtTeach.Text + "," +
        TxtSourceTile.Text;

    for (int i = 0; i < lbBackpack.Items.Count; i++)
    {
        if (i > 0)
        {
            Backpack += ",";
        }
        Backpack += lbBackpack.Items[i];
    }
    OKPressed = true;
    Close();
}
}
}

```

Most of the code was copied from the CharacterForm so I will just go over the new code. In the constructor I loop over all of the items in the backpack for the merchant. I add the item

name appended with the count separated by a colon to the list box for items.

The event handler for the Click event of BtnAddItem adds the concatenates the Text property of TxtItem, a colon and the Text property of TxtCount and adds it to lbBackpack.

In the event handler for the Click event of BtnRemoveItem I check to see if the SelectedIndex property of the backpack list box is greater than or equal to zero. If it is I remove that item from the list box.

In the Click event handler for the OK button after creating the base character string I loop over all of the items in lbBackpack. If i is greater than zero I append a comma to the string Backpack. I then append the item from lbBackpack to the Backpack string. Like before OKPressed is set to true like before and the form is closed.

I added a form that lists the merchants on the map. It looks identical and works identical to the CharacterListForm. There are just some minor code changes. Right click the ShadowEditor project in the Solution Explorer, select Add and then Form (Windows Forms). Name this new form MerchantListForm. Open the MerchantListForm.Designer.cs file. Replace the contents with the following.

```
namespace ShadowEditor
{
    partial class MerchantListForm
    {
        /// <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.BtnDelete = new System.Windows.Forms.Button();
            this.BtnEdit = new System.Windows.Forms.Button();
            this.BtnAdd = new System.Windows.Forms.Button();
            this.LBCharacters = new System.Windows.Forms.ListBox();
            this.SuspendLayout();
        }
    }
}
```



```

//
// BtnDelete
//
this.BtnDelete.Location = new System.Drawing.Point(306, 71);
this.BtnDelete.Name = "BtnDelete";
this.BtnDelete.Size = new System.Drawing.Size(75, 23);
this.BtnDelete.TabIndex = 7;
this.BtnDelete.Text = "Delete";
this.BtnDelete.UseVisualStyleBackColor = true;
this.BtnDelete.Click += new System.EventHandler(this.BtnDelete_Click);
//
// BtnEdit
//
this.BtnEdit.Location = new System.Drawing.Point(306, 42);
this.BtnEdit.Name = "BtnEdit";
this.BtnEdit.Size = new System.Drawing.Size(75, 23);
this.BtnEdit.TabIndex = 6;
this.BtnEdit.Text = "Edit";
this.BtnEdit.UseVisualStyleBackColor = true;
this.BtnEdit.Click += new System.EventHandler(this.BtnEdit_Click);
//
// BtnAdd
//
this.BtnAdd.Location = new System.Drawing.Point(305, 12);
this.BtnAdd.Name = "BtnAdd";
this.BtnAdd.Size = new System.Drawing.Size(75, 23);
this.BtnAdd.TabIndex = 5;
this.BtnAdd.Text = "Add";
this.BtnAdd.UseVisualStyleBackColor = true;
this.BtnAdd.Click += new System.EventHandler(this.BtnAdd_Click);
//
// LBCharacters
//
this.LBCharacters.FormattingEnabled = true;
this.LBCharacters.Location = new System.Drawing.Point(12, 12);
this.LBCharacters.Name = "LBCharacters";
this.LBCharacters.Size = new System.Drawing.Size(287, 277);
this.LBCharacters.TabIndex = 4;
//
// FormMerchantList
//
this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
this.ClientSize = new System.Drawing.Size(407, 309);
this.Controls.Add(this.BtnDelete);
this.Controls.Add(this.BtnEdit);
this.Controls.Add(this.BtnAdd);
this.Controls.Add(this.LBCharacters);
this.Name = "FormMerchantList";
this.Text = "FormMerchantList";
this.Load += new System.EventHandler(this.FormMerchantList_Load);
this.ResumeLayout(false);
}

```

#endregion

```

private System.Windows.Forms.Button BtnDelete;
private System.Windows.Forms.Button BtnEdit;
private System.Windows.Forms.Button BtnAdd;

```

```

        private System.Windows.Forms.ListBox LBCharacters;
    }
}

```

This is all generated code so I'm not going to go over it. Open the code for the form by selecting it in the Solution Explorer and hitting F7 or right click the form in the Solution Explorer and select View Code. Replace the generated code with the following code.

```

using ShadowMonsters.Characters;
using ShadowMonsters.TileEngine;
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 ShadowEditor
{
    public partial class MerchantListForm : Form
    {
        private readonly TileMap map;
        private readonly Editor game1;

        public MerchantListForm(TileMap map, Editor game1)
        {
            InitializeComponent();
            this.map = map;
            this.game1 = game1;
        }

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

            if (frm.OKPressed)
            {
                Character c = Merchant.FromString(game1, frm.Character);
                map.CharacterLayer.Characters.Add(c.SourceTile, c);
                LBCharacters.Items.Add(c.Name);

                string[] parts = frm.Backpack.Split(',');

                foreach (var p in parts)
                {
                    string[] item = p.Split(':');
                    ((Merchant)c).Backpack.AddItem(item[0], int.Parse(item[1]));
                }
            }
        }

        private void BtnEdit_Click(object sender, EventArgs e)
        {
            if (LBCharacters.SelectedIndex < 0 || LBCharacters.Items.Count == 0)

```

```

    {
        return;
    }

    if (LBCharacters.SelectedItem.ToString() == "Unknown")
    {
        return;
    }

    Merchant c =
(Merchant)map.CharacterLayer.GetCharacter(LBCharacters.SelectedItem.ToString());

    MerchantForm formCharacter = new MerchantForm(c);

    formCharacter.TxtName.Enabled = false;
    formCharacter.ShowDialog();

    if (formCharacter.OKPressed)
    {
        Merchant m = Merchant.FromString(game1, formCharacter.Character);
        map.CharacterLayer.Characters[m.SourceTile] = m;

        string[] parts = formCharacter.Backpack.Split(',');

        m.Backpack.Items.Clear();

        foreach (var p in parts)
        {
            string[] item = p.Split(':');
            m.Backpack.AddItem(item[0], int.Parse(item[1]));
        }
    }
}

private void BtnDelete_Click(object sender, EventArgs e)
{
    if (LBCharacters.SelectedIndex < 0)
    {
        return;
    }

    map.CharacterLayer.Characters.Remove(
        map.CharacterLayer.GetCharacter(
            LBCharacters.SelectedItem.ToString()).SourceTile);
    LBCharacters.Items.RemoveAt(LBCharacters.SelectedIndex);
}

private void FormMerchantList_Load(object sender, EventArgs e)
{
    foreach (var c in map.CharacterLayer.Characters.Values)
    {
        if (c is Merchant)
        {
            if (c.Name != null)
            {
                LBCharacters.Items.Add(c.Name);
            }
            else
            {

```



```

        if (!IsActive)
            return;

        if (GamePad.GetState(PlayerIndex.One).Buttons.Back == ButtonState.Pressed
|| Keyboard.GetState().IsKeyDown(Keys.Escape))
            Exit();

        frameCount++;

        if (Xin.CheckKeyReleased(Keys.N) && frameCount > 5)
        {
            NewMapForm form = new NewMapForm(GraphicsDevice);

            form.ShowDialog();

            if (form.OkPressed)
            {
                map = form.TileMap;
                pbTileset.Image = map.TileSet.Textures[0];
                pbPreview.Image = map.TileSet.Textures[0];

                lbTileSets.Items.Clear();
                foreach (string s in map.TileSet.TextureNames)
                    lbTileSets.Items.Add(s);

                lbTileSets.SelectedIndex = 0;
                pbPreview.SourceRectangle =
                    map.TileSet.SourceRectangles[0];
                pbTileset.SourceRectangle = new Rectangle(
                    0,
                    0,
                    map.TileSet.Textures[0].Width,
                    map.TileSet.Textures[0].Height);
            }

            frameCount = 0;
        }

        Vector2 position = new Vector2
        {
            X = Xin.MouseAsPoint.X + camera.Position.X,
            Y = Xin.MouseAsPoint.Y + camera.Position.Y
        };

        tile = Engine.VectorToCell(position);

        if (map != null && viewPort.Contains(Xin.MouseAsPoint))
        {
            if (Xin.MouseState.LeftButton == ButtonState.Pressed)
            {
                switch (lbLayers.SelectedIndex)
                {
                    case 0:
                        map.SetGroundTile(tile.X, tile.Y, lbTileSets.SelectedIndex,
selectedTile);
                        break;
                    case 1:
                        map.SetEdgeTile(tile.X, tile.Y, lbTileSets.SelectedIndex,
selectedTile);
                        break;
                    case 2:
                        map.SetDecorationTile(tile.X, tile.Y,

```

```

lbTileSets.SelectedIndex, selectedTile);
        break;
    case 3:
        map.SetBuildingTile(tile.X, tile.Y,
lbTileSets.SelectedIndex, selectedTile);
        break;
    }
}

if (Xin.MouseState.RightButton == ButtonState.Pressed)
{
    switch (lbLayers.SelectedIndex)
    {
        case 0:
            map.SetGroundTile(tile.X, tile.Y, -1, -1);
            break;
        case 1:
            map.SetEdgeTile(tile.X, tile.Y, -1, -1);
            break;
        case 2:
            map.SetDecorationTile(tile.X, tile.Y, -1, -1);
            break;
        case 3:
            map.SetBuildingTile(tile.X, tile.Y, -1, -1);
            break;
    }
}

if (pbTileset != null &&
    pbTileset.DestinationRectangle.Contains(
        Xin.MouseAsPoint) &&
    Xin.CheckMouseReleased(MouseButtons.Left))
{
    Point previewPoint = new Point(
        Xin.MouseAsPoint.X - pbTileset.DestinationRectangle.X,
        Xin.MouseAsPoint.Y - pbTileset.DestinationRectangle.Y);
    float xScale =
(float)map.TileSet.Textures[lbTileSets.SelectedIndex].Width /
        pbTileset.DestinationRectangle.Width;

    float yScale =
(float)map.TileSet.Textures[lbTileSets.SelectedIndex].Height /
        pbTileset.DestinationRectangle.Height;

    Point tilesetPoint = new Point(
        (int)(previewPoint.X * xScale),
        (int)(previewPoint.Y * yScale));

    Point clickedTile = new Point(
        tilesetPoint.X / map.TileSet.TileWidth,
        tilesetPoint.Y / map.TileSet.TileHeight);

    selectedTile = clickedTile.Y * map.TileSet.TilesWide + clickedTile.X;
    pbPreview.SourceRectangle =
        map.TileSet.SourceRectangles[selectedTile];
}

if (Xin.CheckKeyReleased(Keys.C) && frameCount > 5 && map != null)
{
    CharacterListForm frm = new CharacterListForm(map, this);
    frm.ShowDialog();
}

```

```

if (Xin.CheckKeyReleased(Keys.M) && frameCount > 5 && map != null)
{
    MerchantListForm frm = new MerchantListForm(map, this);
    frm.ShowDialog();
}

if (Xin.CheckKeyReleased(Keys.F1) && frameCount > 5)
{
    WF.SaveFileDialog sfd = new WF.SaveFileDialog();
    sfd.Filter = "Tile Map (*.map)|*.map";
    WF.DialogResult result = sfd.ShowDialog();

    if (result == WF.DialogResult.OK)
    {
        SaveMap(sfd.FileName);
    }
}

if (Xin.CheckKeyReleased(Keys.F2) && frameCount > 5)
{
    WF.OpenFileDialog ofd = new WF.OpenFileDialog();
    ofd.Filter = "Tile Map (*.map)|*.map";
    WF.DialogResult result = ofd.ShowDialog();

    if (result == WF.DialogResult.OK)
    {
        LoadMap(ofd.FileName);

        pbTileset.Image = map.TileSet.Textures[0];
        pbPreview.Image = map.TileSet.Textures[0];

        lbTileSets.Items.Clear();
        foreach (string s in map.TileSet.TextureNames)
            lbTileSets.Items.Add(s);

        lbTileSets.SelectedIndex = 0;
        pbPreview.SourceRectangle =
            map.TileSet.SourceRectangles[0];
        pbTileset.SourceRectangle = new Rectangle(
            0,
            0,
            map.TileSet.Textures[0].Width,
            map.TileSet.Textures[0].Height);
    }
}

HandleScrollMap();
controls.Update(gameTime);
base.Update(gameTime);
}

```

The new code checks to see if the C key has been released. If it has it creates a CharacterListForm and then calls the ShowDialog method on the form. There is no need to process the return from the form because the changes are made to the CharacterLayer directly. It also checks to see if the M key has been release. If it has been release it creates a MerchantListForm and then calls the ShowDialog method. Again, we don't have to do anything because the changes are made to the CharacterLayer directly.

I need to make a change to the Game1 class in the ShadowMonsters project. I moved the

code for creating the animations out of the Initialize method and placed it in a method of its own. Replace the Initialize method with the following and add the BuildAnimations method.

```
protected override void Initialize()
{
    // TODO: Add your initialization logic here

    BuildAnimations();

    Components.Add(new Xin(this));
    Components.Add(new FontManager(this));

    Game1.Player = new Player(this, "Bonnie", true, @"Sprites\mage_f");
    base.Initialize();
}

public static void BuildAnimations()
{
    Animation animation = new Animation(3, 32, 36, 0, 0);
    animations.Add(AnimationKey.WalkUp, animation);

    animation = new Animation(3, 32, 36, 0, 36);
    animations.Add(AnimationKey.WalkRight, animation);

    animation = new Animation(3, 32, 36, 0, 72);
    animations.Add(AnimationKey.WalkDown, animation);

    animation = new Animation(3, 32, 36, 0, 108);
    animations.Add(AnimationKey.WalkLeft, animation);
}
```

The last thing to do is call the BuildAnimations method in the Editor class. I did that in the Initialize method. Update that method to the following.

```
protected override void Initialize()
{
    // TODO: Add your initialization logic here
    Components.Add(new Xin(this));
    Game1.BuildAnimations();

    base.Initialize();
}
```

If you build and run the project now you can create characters and merchants on the map. I'm going to wrap this tutorial up here. I will be starting work on the next tutorial shortly. Keep checking back on the blog for news on that tutorial. I hope to have it up in the next week or so.

I wish You the best in Your MonoGame Programming Adventures!