Shadow Monsters – MonoGame Tutorial Series
Chapter 20
Improvements And Sound

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. 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 I'm going to make some improvements to the game. The first thing that I'm going to do is add tile based collision. We can paint collision but the game doesn't support it. Open the GamePlayState and update the HandleMovement method to the following.

```
private void HandleMovement(GameTime gameTime)
{
    if (Xin.KeyboardState.IsKeyDown(Keys.W) && !inMotion)
    {
        motion.Y = -1;
        Game1.Player.Sprite.CurrentAnimation = AnimationKey.WalkUp;
        Game1.Player.Sprite.IsAnimating = true;
        inMotion = true;
        collision = new Rectangle(
            (int)Game1.Player.Sprite.Position.X,
            (int)Game1.Player.Sprite.Position.Y - Engine.TileHeight * 2,
            Engine.TileWidth,
            Engine.TileHeight);
    }
    else if (Xin.KeyboardState.IsKeyDown(Keys.S) && !inMotion)
    {
        motion.Y = 1;
        Game1.Player.Sprite.CurrentAnimation = AnimationKey.WalkDown;
        Game1.Player.Sprite.IsAnimating = true;
        inMotion = true;
        collision = new Rectangle(
            (int)Game1.Player.Sprite.Position.X,
            (int)Game1.Player.Sprite.Position.Y + Engine.TileHeight * 2,
            Engine.TileWidth,
            Engine.TileHeight);
    }
    else if (Xin.KeyboardState.IsKeyDown(Keys.A) && !inMotion)
    {
        motion.X = -1;
        Game1.Player.Sprite.CurrentAnimation = AnimationKey.WalkLeft;
        Game1.Player.Sprite.IsAnimating = true;
        inMotion = true;
        collision = new Rectangle(
            (int)Game1.Player.Sprite.Position.X - Engine.TileWidth * 2,
            (int)Game1.Player.Sprite.Position.Y,
            Engine.TileWidth,
            Engine.TileHeight);
    }
    else if (Xin.KeyboardState.IsKeyDown(Keys.D) && !inMotion)
    {
        motion.X = 1;
        Game1.Player.Sprite.CurrentAnimation = AnimationKey.WalkRight;
        Game1.Player.Sprite.IsAnimating = true;
        inMotion = true;
        collision = new Rectangle(
            (int)Game1.Player.Sprite.Position.X + Engine.TileWidth * 2,
            (int)Game1.Player.Sprite.Position.Y,
            Engine.TileWidth,
            Engine.TileHeight);
    }

    if (motion != Vector2.Zero)
    {
        motion.Normalize();
        motion *=
            (Game1.Player.Sprite.Speed *
            (float)gameTime.ElapsedGameTime.TotalSeconds
            * Settings.Multiplier);
        Rectangle pRect = new Rectangle(
                (int)(Game1.Player.Sprite.Position.X + motion.X),
                (int)(Game1.Player.Sprite.Position.Y + motion.Y),
                Engine.TileWidth,
                Engine.TileHeight);

        if (pRect.Intersects(collision))
        {
            Game1.Player.Sprite.IsAnimating = false;
            inMotion = false;
            motion = Vector2.Zero;
        }

        foreach (Point p in world.Map.CharacterLayer.Characters.Keys)
        {
            Rectangle r = new Rectangle(
                p.X * Engine.TileWidth,
                p.Y * Engine.TileHeight,
                Engine.TileWidth,
                Engine.TileHeight);

            if (r.Intersects(pRect))
            {
                motion = Vector2.Zero;
                Game1.Player.Sprite.IsAnimating = false;
                inMotion = false;
            }
        }

        foreach (Point p in world.Map.CollisionLayer.Collisions.Keys)
```

```
            {
                Rectangle r = new Rectangle(
                    p.X * Engine.TileWidth,
                    p.Y * Engine.TileHeight,
                    Engine.TileWidth,
                    Engine.TileHeight);

                if (r.Intersects(pRect))
                {
                    motion = Vector2.Zero;
                    Game1.Player.Sprite.IsAnimating = false;
                    inMotion = false;
                }
            }
            Vector2 newPosition = Game1.Player.Sprite.Position + motion;
            newPosition.X = (int)newPosition.X;
            newPosition.Y = (int)newPosition.Y;

            Game1.Player.Sprite.Position = newPosition;
            motion = Game1.Player.Sprite.LockToMap(
                new Point(
                    world.Map.WidthInPixels,
                    world.Map.HeightInPixels),
                motion);

            if (motion == Vector2.Zero)
            {
                Vector2 origin = new Vector2(
                        Game1.Player.Sprite.Position.X + Game1.Player.Sprite.Origin.X,
                        Game1.Player.Sprite.Position.Y + Game1.Player.Sprite.Origin.Y);
                Game1.Player.Sprite.Position = Engine.VectorFromOrigin(origin);
                inMotion = false;
                Game1.Player.Sprite.IsAnimating = false;
            }
        }
    }
```

The new code loops over the keys in the Collisions connection of CollisionLayer of the current map. Using the key, which is a Point, I create a Rectangle using the point and the tile width and tile height from the engine. If the rectangle and the player's rectangle intersect I cancel the requested movement, set the IsAnimating property of the sprite to false and the inMotion variable to false.

I'm going to update the GamePlayState to draw collisions if it is in DEBUG mode but not if it is in RELEASE mode. I did that by creating textures in the LoadContent method and updating the Draw method to use the preprocessor to call the Draw method of the World class passing in true for the optional parameter. Update the LoadContent method and Draw method of the GamePlayState to the following.

```
    protected override void LoadContent()
    {
        Texture2D texture = new Texture2D(GraphicsDevice, 16, 16);
        Color[] buffer = new Color[16 * 16];

        for (int i = 0; i < buffer.Length; i++)
        {
            buffer[i] = new Color(255, 0, 0, 128);
            }

            texture.SetData(buffer);

            CollisionLayer.Texture = texture;

            for (int i = 0; i < buffer.Length; i++)
            {
                buffer[i] = new Color(0, 0, 255, 128);
            }

            texture.SetData(buffer);

            PortalLayer.Texture = texture;
            base.LoadContent();
        }

        public override void Draw(GameTime gameTime)
        {
            base.Draw(gameTime);

#if DEBUG
            world.Draw(gameTime, GameRef.SpriteBatch, Engine.Camera, true);
#else
            world.Draw(gameTmie, GameRef.SpriteBatch, Engine.Camera);
#endif

            GameRef.SpriteBatch.Begin(
                SpriteSortMode.Deferred,
                BlendState.AlphaBlend,
                SamplerState.PointClamp,
                null,
                null,
                null,
                Engine.Camera.Transformation);
            Game1.Player.Draw(gameTime);
            GameRef.SpriteBatch.End();
        }
```

The next thing I want to do is change the tile set and add a couple of collisions to the collision layer to test the collision code. First, the tile set. I found a nice tile set on OpenGameArt.org that I'm using. It is by FuwanekoGames and you can find it at the following URL:

https://opengameart.org/content/tiny16-tileset

Download it to your computer. Open the MonoGame Pipeline Tool. Right click the Tiles folder, select Add and then Existing Item. Browse to the tiny-16.png file and add it to the folder. Save the project, build it then close the MonoGame Pipeline Tool. Update the SetUpNewGame method to the following code.

```
        internal void SetUpNewGame()
        {
            MoveManager.FillMoves();
            ShadowMonsterManager.FromFile(@".\Content\ShadowMonsters.txt", content);

            Game1.Player.AddShadowMonster(ShadowMonsterManager.GetShadowMonster("water1"));
            Game1.Player.SetCurrentShadowMonster(0);
            Game1.Player.BattleShadowMonsters[0] = Game1.Player.GetShadowMonster(0);

            TileSet set = new TileSet(10, 10, 16, 16);

            set.TextureNames.Add("tiny-16");
            set.Textures.Add(content.Load<Texture2D>(@"Tiles\tiny-16"));

            TileLayer groundLayer = new TileLayer(100, 100, 0, 1);
            TileLayer edgeLayer = new TileLayer(100, 100);
            TileLayer buildingLayer = new TileLayer(100, 100);
            TileLayer decorationLayer = new TileLayer(100, 100);

            for (int i = 0; i < 1000; i++)
            {
                decorationLayer.SetTile(random.Next(0, 100), random.Next(0, 100), 0, 0);
            }

            TileMap map = new TileMap(set, groundLayer, edgeLayer, buildingLayer, 
decorationLayer, "level1");

            Character c = Character.FromString(GameRef, 
"Paul,ninja_m,WalkDown,PaulHello,0,2:2,fire1,fire1,,,,,,dark1");
            c.Sprite.Position = new Vector2(
                c.SourceTile.X * Engine.TileWidth,
                c.SourceTile.Y * Engine.TileHeight);

            map.CharacterLayer.Characters.Add(c.SourceTile, c);

            Merchant m = Merchant.FromString(GameRef, 
"Bonnie,ninja_f,WalkLeft,BonnieHello,0,4:4,earth1,earth1,,,,,,");
            m.Sprite.Position = new Vector2(
                m.SourceTile.X * Engine.TileWidth,
                m.SourceTile.Y * Engine.TileHeight);

            m.Backpack.AddItem("Potion", 99);
            m.Backpack.AddItem("Antidote", 10);

            map.CharacterLayer.Characters.Add(m.SourceTile, m);

            map.CollisionLayer.Collisions.Add(new Point(5, 5), CollisionType.Impassable);
            map.CollisionLayer.Collisions.Add(new Point(6, 6), CollisionType.Impassable);

            world = new World(new Portal(new Point(10, 10), new Point(10, 10), map.MapName));
            world.Maps.Add(map.MapName, map);
            world.ChangeMap(map.MapName);
            Game1.Player.Sprite.Position = new Vector2(
                world.StartingMap.SourceTile.X * Engine.TileWidth,
                world.StartingMap.SourceTile.Y * Engine.TileHeight);
        }
```

When creating the TileSet I pass in (10, 10, 16, 16) where 10 is the tiles wide, 10 is the tiles high, 16 is the tile width and 16 is the tile height. I then add the string tiny-16 to the texture names and load the tiny-16 tileset. After creating the merchant I create collisions at tiles (5,5) and (6,6).

I'm going to change the behavior of the Escape key. I don't want to just exit the game when it is pressed. If I'm in the game I want to go back to the menu. If I'm in the menu I will exit the game. In the states where I press the right mouse button to go back I want the escape key to go back. The first thing that I did was remove the code in the Update method that exits the game. Change the Update method of the Game1 class to the following.

```
protected override void Update(GameTime gameTime)
{
    // TODO: Add your update logic here
    base.Update(gameTime);
}
```

Now, update the Update method of the GamePlayState to the following code.

```
public override void Update(GameTime gameTime)
{
    frameCount++;
    HandleMovement(gameTime);

    if (Xin.CheckKeyReleased(Keys.Escape))
    {
        manager.ChangeState(GameRef.MainMenuState);
    }

    if ((Xin.CheckKeyReleased(Keys.Space) ||
        Xin.CheckKeyReleased(Keys.Enter)) && frameCount >= 5)
    {
        frameCount = 0;
        StartInteraction();
        HandlePortals();
    }

    if (Xin.CheckKeyReleased(Keys.I))
    {
        manager.PushState(GameRef.ItemSelectionState);
    }

    if ((Xin.CheckKeyReleased(Keys.B)) && frameCount >= 5)
    {
        frameCount = 0;
        StartBattle();
    }

    if (Xin.CheckKeyReleased(Keys.F1))
    {
        SaveGame();
    }
    if (Xin.CheckKeyReleased(Keys.F2))
    {
        LoadGame();
    }

    Engine.Camera.LockToSprite(
        world.Maps[world.CurrentMapName],
        Game1.Player.Sprite,
        new Rectangle(0, 0, Settings.Resolution.X, Settings.Resolution.Y));
    world.Update(gameTime);
    Game1.Player.Update(gameTime);

    base.Update(gameTime);
}
```

The new code checks to see if the Escape key has been pressed. If it has it changes the game state to MainMenuState. I made a quick change to MoveManager.FillMoves. If it is called twice it will throw an exception. I added a call to Clear to clear the items.

```
        public static void FillMoves()
        {
            allMoves.Clear();
            allMoves.Add("Tackle", new Tackle());
            allMoves.Add("Block", new Block());
        }
```

If you run now and go to the GamePlayState you can go back to the main menu by pressing the Escape key. Now I'm going to exit the game from the main menu if the Escape key is pressed. I also updated the menu so that space activates a menu item instead of just enter and the left mouse button. Update the Update method of the MainMenuState to the following code.

```
        public override void Update(GameTime gameTime)
        {
            frameCount++;

            if (menu.SelectedIndex == -1)
            {
                menu.SetMenuItems(new[] { "New Game", "Continue", "Options", "Exit" });
            }

            if (Xin.CheckKeyReleased(Keys.Escape))
            {
                GameRef.Exit();
            }

            if ((Xin.CheckKeyReleased(Keys.Enter) ||
                Xin.CheckKeyReleased(Keys.Space)
                || (menu.MouseOver && Xin.CheckMouseReleased(MouseButtons.Left)))
                && frameCount > 5)
            {
                switch (menu.SelectedIndex)
                {
                    case 0:
                        manager.PopState();
                        manager.PushState(GameRef.GamePlayState);
                        GameRef.GamePlayState.SetUpNewGame();
                        break;
                    case 1:
                        string path = Environment.GetFolderPath(
                            Environment.SpecialFolder.ApplicationData);
                        path += "\\ShadowMonsters\\ShadowMonsters.sav";

                        if (!File.Exists(path))
                        {
                            return;
                        }

                        MoveManager.FillMoves();
                        ShadowMonsterManager.FromFile(@".\Content\ShadowMonsters.txt",
content);
                        manager.PopState();
                        manager.PushState(GameRef.GamePlayState);
                        GameRef.GamePlayState.LoadGame();
                        GameRef.GamePlayState.ResetEngine();
                        break;
                    case 2:
                        manager.PushState(GameRef.OptionState);
                        break;
                    case 3:
                        GameRef.Exit();
                        break;
                }
                frameCount = 0;
            }
            base.Update(gameTime);
        }
```

I'm going to add in some music and sound effects now. First, to download the music. Go to https://freemusicarchive.org/music/Augustin_C/Fantasy_Music/Enchanted_Village_1062. Right click the download arrow and select Save link as to download the file. Go to Visual Studio and open the MonoGame Pipeline tool. Right click the Content node, select Add and then New Folder. Name this new folder Music. Right click the Music folder and select Add then Existing Item. Browse for and add the Augustin_C_-_22_-_Enchanted_Village.mp3 file. If prompted select Copy the file to the directory option. Save and build the project.

In MonoGame there are are three class that I use for sounds. The first is Song. A song is usually an MP3. It is loaded using the ContentManager. A song is played using the MediaPlayer, the second class. The third class is SoundEffects. Sound effects are played immediately on their own and do not use the MediaPlayer.

Now I'm going to add a class to manage music and sound effects in the game. Right click the Components folder under ShadowMonsters, select Add and then Class. Name this new class Muse. Here is the code.

```
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Audio;
using Microsoft.Xna.Framework.Media;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ShadowMonsters.Components
{
    public class Muse : GameComponent
    {
        public static Dictionary<string, Song> Songs { get; private set; }
        public static Dictionary<string, SoundEffect> SoundEffects { get; private set;}
        private static Song Song { get; set; }

        public Muse(Game game)
            : base(game)
        {
            Songs = new Dictionary<string, Song>();
            SoundEffects = new Dictionary<string, SoundEffect>();
        }

        public static void PlaySong(string name)
        {
            if (Songs.ContainsKey(name))
            {
                Song = Songs[name];
                MediaPlayer.Play(Song);
            }
        }

        public static void StopSong()
        {
            MediaPlayer.Stop();
        }

        public static void PlaySoundEffect(string name)
        {
            if (SoundEffects.ContainsKey(name))
            {
                SoundEffects[name].Play();
            }
        }

        public static void SetSongVolume(float volume)
        {
            MediaPlayer.Volume = MathHelper.Clamp(volume, 0f, 1f);
        }

        public static void SetEffectVolume(float volume)
        {
            SoundEffect.MasterVolume = MathHelper.Clamp(volume, 0f, 1f);
        }
    }
}
```

This is a simple class. It has some static properties that are public get and private set. One is a Dictionary<string, Song> that holds the songs and the other is a Dictionary<string, SoundEffect> that holds the sound effect. There is a private property that is a Song that holds the currently playing song, if any. The constructor initializes the dictionaries. The PlaySong method takes a string parameter. If the Songs property contains the key I set the Song field to the song and call MediaPlayer.Play passing in the song to start play back. The StopSong method calls MediaPlayer.Stop to stop playing the current song. There is a method PlaySoundEffect that takes a string parameter. It checks if the key exists in the SoundEffects collection. If it does it calls Play to play it. There is a public method SetSongVolume that sets the volume at which music will play. It does that by setting the Volume property of MediaPlayer. There is also a public method SetEffectVolume that sets the value at which sound effects will play. That is done by setting the MasterVolume property of the SoundEffect class. The volumes are clamped between 0 and 1.

Now let's add the component in the Game1 class. While I'm updating it I will update it so that settings will be preserved between sessions. Add the following using statement and then update the constructor to the following code.

```
using ShadowMonsters.Components;

        public Game1()
        {
            Settings.Load();
            graphics = new GraphicsDeviceManager(this)
            {
                PreferredBackBufferWidth = Settings.Resolution.X,
                PreferredBackBufferHeight = Settings.Resolution.Y
            };.

            graphics.ApplyChanges();


            foreach (var v in graphics.GraphicsDevice.Adapter.SupportedDisplayModes)
            {
                Point p = new Point(v.Width, v.Height);
                string s = v.Width + " by " + v.Height + " pixels";

                if (v.Width >= 1280 && v.Height >= 720)
                {
                    Resolutions.Add(s, p);
                }
            }

            Content.RootDirectory = "Content";

            stateManager = new GameStateManager(this);
            Components.Add(stateManager);

            Components.Add(new Muse(this));
            Muse.SetEffectVolume(Settings.SoundVolume);
            Muse.SetSongVolume(Settings.MusicVolume);

            gamePlayState = new GamePlayState(this);
            conversationState = new ConversationState(this);
            levelUpState = new LevelUpState(this);
            damageState = new DamageState(this);
            battleOverState = new BattleOverState(this);
            battleState = new BattleState(this);
            actionSelectionState = new ActionSelectionState(this);
            shadowMonsterSelectionState = new ShadowMonsterSelectionState(this);
            startBattleState = new StartBattleState(this);
            shopState = new ShopState(this);
            itemSelectionState = new ItemSelectionState(this);
            useItemState = new UseItemState(this);
            mainMenuState = new MainMenuState(this);
            optionState = new OptionState(this);

            stateManager.PushState(mainMenuState);
            ConversationManager.Instance.CreateConversations(this);
            IsMouseVisible = true;
        }
```

The first thing the constructor does now is call Settings.Load to load the game settings. When setting the width and height of the window I now use Settings.Resolution to set the properties instead of the hard coded 1280 by 720. After adding Muse to the game components I call SetEffectValue passing in Settings.SoundVolume. For songs I call SetSongVolume passing in Settings.MusicVolume.

Let's load the song that we just downloaded in the game and play it on the MainMenuState and stop it if the player goes to the GamePlayState. Add the following using statement and update the LoadContent and Update methods of the MainMenuState to the following.

```
using Microsoft.Xna.Framework.Media;

        protected override void LoadContent()
        {
            base.LoadContent();
            Texture2D texture = new Texture2D(GraphicsDevice, 400, 50);

            Color[] buffer = new Color[400 * 50];

            for (int i = 0; i < 400 * 50; i++)
                buffer[i] = Color.Black;

            texture.SetData(buffer);

            menu = new MenuComponent(GameRef, texture);

            if (!Muse.Songs.ContainsKey("main_menu"))
            {
                Muse.Songs.Add(
                    "main_menu",
                    content.Load<Song>(@"Music\Augustin_C_-_22_-_Enchanted_Village"));
                Muse.PlaySong("main_menu");
                MediaPlayer.IsRepeating = true;
            }

            Components.Add(menu);
        }

        public override void Update(GameTime gameTime)
        {
            frameCount++;

            if (menu.SelectedIndex == -1)
            {
                menu.SetMenuItems(new[] { "New Game", "Continue", "Options", "Exit" });
            }

            if (Xin.CheckKeyReleased(Keys.Escape))
            {
                GameRef.Exit();
            }

            if (Xin.CheckKeyReleased(Keys.Enter) ||
                Xin.CheckKeyReleased(Keys.Space)
                || (menu.MouseOver && Xin.CheckMouseReleased(MouseButtons.Left)) && frameCount
> 5)
            {
                switch (menu.SelectedIndex)
                {
                    case 0:
                        manager.PopState();
                        manager.PushState(GameRef.GamePlayState);
                        GameRef.GamePlayState.SetUpNewGame();
                        Muse.StopSong();
                        break;
                    case 1:
                        string path = Environment.GetFolderPath(
                            Environment.SpecialFolder.ApplicationData);
                        path += "\\ShadowMonsters\\ShadowMonsters.sav";

                        if (!File.Exists(path))
                        {
                            return;
                        }

                        Muse.StopSong();
                        MoveManager.FillMoves();
                        ShadowMonsterManager.FromFile(@".\Content\ShadowMonsters.txt",
content);
                        manager.PopState();
                        manager.PushState(GameRef.GamePlayState);
                        GameRef.GamePlayState.LoadGame();
                        GameRef.GamePlayState.ResetEngine();
                        break;
                    case 2:
                        manager.PushState(GameRef.OptionState);
                        break;
                    case 3:
                        GameRef.Exit();
                        break;
                }
                frameCount = 0;
            }
            base.Update(gameTime);
        }
```

The using statement for Microsoft.Xna.Framework.Media brings the Song and MediaPlayer classes into scope so we don't have to fully qualify them. In the LoadContent method I check to see if the key exists in the Songs property of Muse. If it does not exist I add the key with the Song using the Content.Load method. I then call the PlaySong method passing in the key. I set the IsRepeating of the MediaPlayer to true so the song will repeat. In the Update method I modified the cases for 0 and 1 to call MediaPlayer.Stop to stop play back of the song.

Let's add a sound effect. I downloaded one by NenadSimic from OpenGameArt.org. You can find it here https://opengameart.org/content/menu-selection-click. Download the sound effect. Open the MonoGame Pipeline Tool. Right click the Content folder, select Add and then New Folder. Name this new folder SoundEffects. Right click the SoundEffects folder, select Add then Existing Item. Browse to the Menu Selection Click.wav file. When prompted select the Copy the file to the directory option. Save and build the project.

Add the following using statement to the Game1 class and update the LoadContent method to the following.

```
using Microsoft.Xna.Framework.Audio;

        protected override void LoadContent()
        {
            // Create a new SpriteBatch, which can be used to draw textures.
            spriteBatch = new SpriteBatch(GraphicsDevice);

            Muse.SoundEffects.Add(
                "menu_click",
                Content.Load<SoundEffect>(@"SoundEffects\Menu Selection Click"));
        }
```

The LoadContent method adds a SoundEffect with the key menu_click to the SoundEffects collection of Muse. Let's play this when a button is clicked, the left right selector is clicked or a menu item on the main is selected. Add the following using statement and change the OnClick method of the Button class to the following.

```
using ShadowMonsters.Components;

        private void OnClick()
        {
            Muse.PlaySoundEffect("menu_click");
            Click?.Invoke(this, null);
        }
```

The using statement brings Muse into scope. In the OnClick method I just call the PlaySoundEffect method of Muse passing in the name of the sound effect.

Add the following using statement and change the OnSelectionChanged method of the LeftRightSelector to the following.

```
using ShadowMonsters.Components;

        protected void OnSelectionChanged()
        {
            Muse.PlaySoundEffect("menu_click");
            SelectionChanged?.Invoke(this, null);
        }
```

The using statement brings Muse into scope. In the OnSelectionChanged method I just call the PlaySoundEffect method of Muse passing in the name of the sound effect.

In the Update method of the MainMenuSta I where check if space, enter or left mouse button has been released I call Muse.PlaySoundEffect to play the click sound effect. Change the Update method of the MainMenuState to the following.

```
        public override void Update(GameTime gameTime)
        {
            frameCount++;

            if (menu.SelectedIndex == -1)
            {
                menu.SetMenuItems(new[] { "New Game", "Continue", "Options", "Exit" });
            }

            if (Xin.CheckKeyReleased(Keys.Escape))
            {
                GameRef.Exit();
            }

            if (Xin.CheckKeyReleased(Keys.Enter) ||
                Xin.CheckKeyReleased(Keys.Space)
                || (menu.MouseOver && Xin.CheckMouseReleased(MouseButtons.Left)) && frameCount
> 5)
            {
                Muse.PlaySoundEffect("menu_click");
                switch (menu.SelectedIndex)
                {
                    case 0:
                        manager.PopState();
                        manager.PushState(GameRef.GamePlayState);
                        GameRef.GamePlayState.SetUpNewGame();
                        Muse.StopSong();
                        break;
                    case 1:
                        string path = Environment.GetFolderPath(
                            Environment.SpecialFolder.ApplicationData);
                        path += "\\ShadowMonsters\\ShadowMonsters.sav";

                        if (!File.Exists(path))
                        {
                            return;
                        }

                        Muse.StopSong();
                        MoveManager.FillMoves();
                        ShadowMonsterManager.FromFile(@".\Content\ShadowMonsters.txt",
content);
                        manager.PopState();
                        manager.PushState(GameRef.GamePlayState);
                        GameRef.GamePlayState.LoadGame();
                        GameRef.GamePlayState.ResetEngine();
                        break;
                    case 2:
                        manager.PushState(GameRef.OptionState);
                        break;
                    case 3:
                        GameRef.Exit();
                        break;
                }
                frameCount = 0;
            }
            base.Update(gameTime);
        }
```

I am 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!