

Three Thing Game

www.threethinggame.com

 UNIVERSITY OF Hull

www.threethinggame.com

UNIVERSITY OF Hull


Rob Miles
Department of Computer Science

Agenda

- Getting started writing an XNA game
- Getting input
- Writing Text
- Making sounds

XNA

- XNA is a framework for writing games
- Includes a set of professional tools for game production and content management
- It works within Visual Studio
 - There are XNA project types in the same way we have Console project types

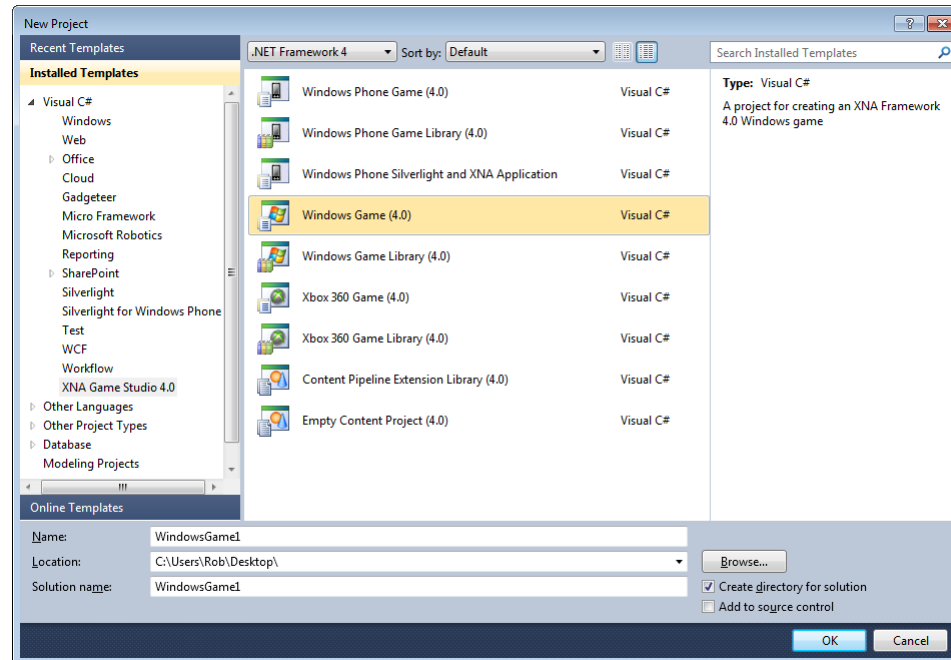
Running Games

- Games can be run on the PC, Xbox 360 or Windows Phone 7.5 device
- There is an Open Source version of XNA called MonoGame that will run on Windows 8, Windows Phone 8. Android, IOS and PlayStation Vita
- You can use this for your Three Thing Game project, but you can use any other framework if you prefer

XNA Versions

- The latest version of XNA is 4.0
 - This works with Visual Studio 2010
- You can obtain this from <http://create.msdn.com>
- This also contains the development environment for Windows Phone and Xbox 360
- Not all installations of Visual Studio on campus have the XNA components installed
- The Fenner Computer Suite and the labs in the Robert Blackburn Building have XNA

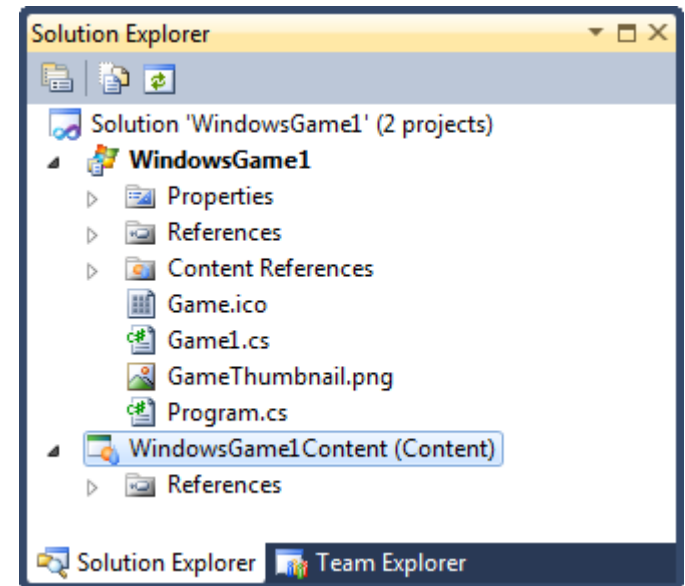
Creating a Game



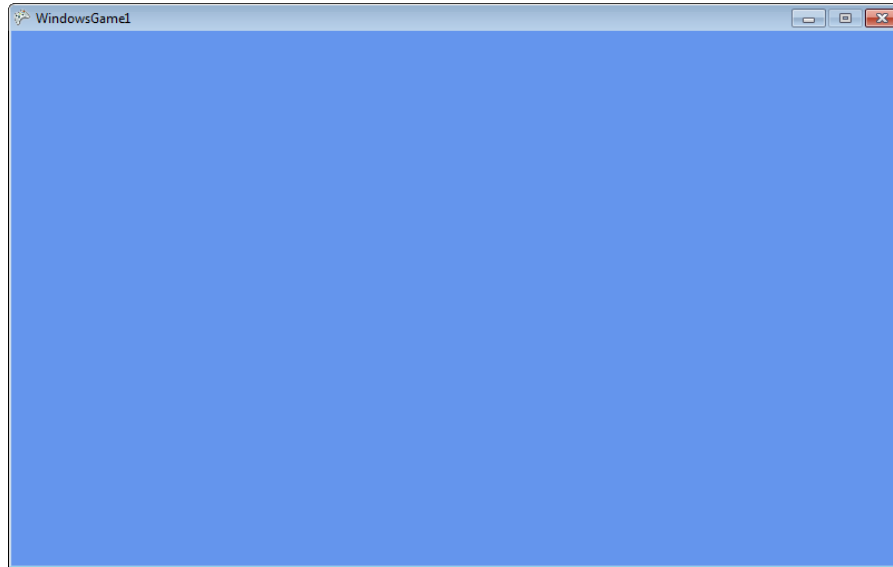
- Visual Studio in the Fenner Lab has XNA 4.0 installed
- You need to create a Windows Game

The Game Project

- The solution explorer shows the items that make up our game project
- At the moment there are a couple of class files which are created automatically
- The solution will also contain any content that we add to the game project



Empty Game Display



- At the moment all our empty game does is display a blue screen
- This is because the behaviour of the Draw method in a brand new project is to clear the screen to blue

How Games Work

- Every game that has ever been written has these fundamental behaviours:
- Initialise all the resources at the start
 - fetch all textures, models, scripts etc
- Repeatedly run the game loop:
 - Update the game world
 - read the controllers, update the state and position of game elements
 - Draw the game world
 - render the game elements on the viewing device

Methods in an XNA game

- The XNA Game class contains methods that will provide these behaviours
- Initialise all the resources at the start
 - The `Initialize` and `LoadContent` methods
- Repeatedly run the game loop:
 - Update the game world
 - The `Update` method
 - Draw the game world
 - The `Draw` method

Getting Started with XNA

- When you create a new XNA game project you are provided with empty versions of the game methods
- Creating an XNA game is a matter of filling in these methods to get the game behaviours that are required
- We are going to start by getting some clouds moving around the display
- Then we are going to add some complication and see where it gets us

Cloud and Games

- Apparently the future of computing is “in the cloud”
- Perhaps the future of games is too
- We can start with a drawing of a cloud and see where this takes us
- For me this is the fun of making games



Creating a Game World

```
// Game World  
  
Texture2D cloudTexture;  
  
Vector2 cloudPosition;
```

- A game needs a “Game World” which holds all the objects in the game
 - **LoadContent** will put content into them
 - **Update** will update their state
 - **Draw** will draw them
- To start with our game just contains a cloud texture and position

Loading the Cloud Texture

```
protected override void LoadContent()  
{  
    spriteBatch = new SpriteBatch(GraphicsDevice);  
    cloudPosition = new Vector2(0, 0);  
    cloudTexture = Content.Load<Texture2D>("Cloud");  
}
```

- This code loads our cloud texture
- It also sets the draw position for the cloud
- It also makes a SpriteBatch, which is used by XNA to batch up drawing operations

Drawing the Cloud Texture

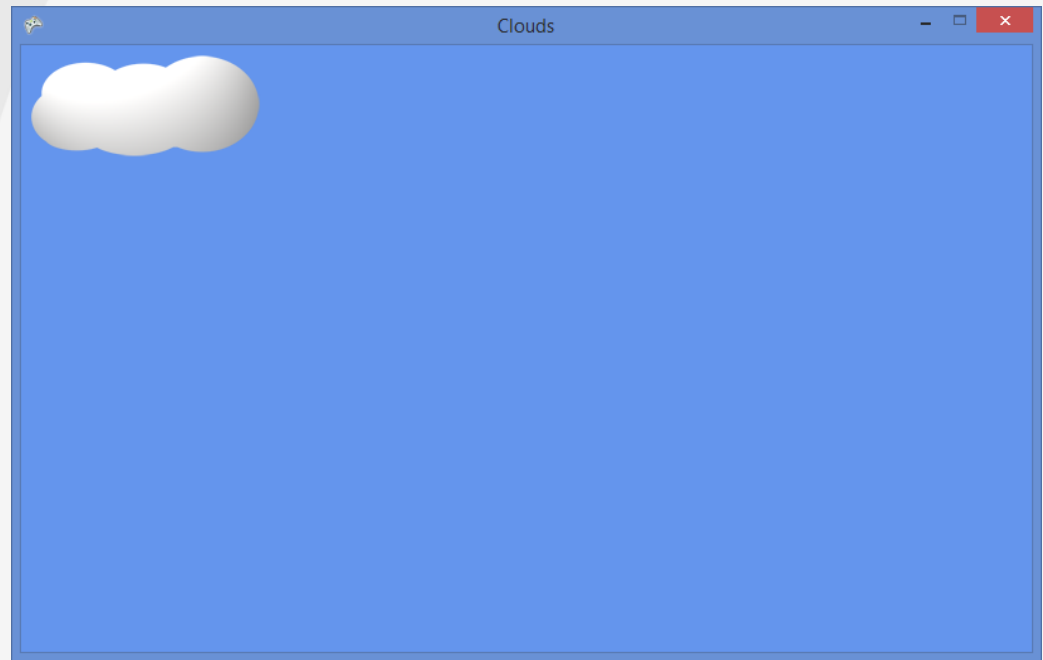
```
protected override void Draw(GameTime gameTime)
{
    GraphicsDevice.Clear(Color.CornflowerBlue);
    spriteBatch.Begin();
    spriteBatch.Draw(cloudTexture, cloudPosition,
                    Color.White);

    spriteBatch.End();
    base.Draw(gameTime);
}
```

- This code uses the spriteBatch to draw our cloud
- It also draws a blue sky as a background

1: Simple Cloud

Rob Miles
University of Hull



demo

Adding Movement

- A cloud that doesn't move is rather boring
- We need to make it do things
 - Perhaps it could drift across the screen
- To do this we use the Update method
- This is called by XNA 60 times a second to update the position of elements in the game

Drifting our Cloud

```
Vector2 cloudSpeed = new Vector2(1.5f, 0);  
  
protected override void Update(GameTime gameTime)  
{  
    cloudPosition += cloudSpeed;  
  
    base.Update(gameTime);  
}
```

- The Update method is called 60 times a second
- We can use it to drift the cloud across the screen
- Note that XNA provides a Vector type which we are using to position the sprite
- We can also perform Vector arithmetic to move it around

2: Drifting Cloud

Rob Miles
University of Hull



demo

Creating Game Components

```
public interface ISprite
{
    void Draw(CloudGame game);
    void Update(CloudGame game);
}

// Game World
List<ISprite> gameSprites = new List<ISprite>();
```

- We really need more than one cloud
- We can make a ISprite component which has Draw and Update behaviours
- We can then make a list of these to use in our game

An overview of the Cloud Class

```
class Cloud : CloudGame.ISprite
{
    public Texture2D CloudTexture;
    public Vector2 CloudPosition;
    public Vector2 CloudSpeed;

    public void Draw(CloudGame game) ...

    public void Update(CloudGame game) ...

    public Cloud(Texture2D inTexture,
                 Vector2 inPosition,
                 Vector2 inSpeed) ...
}
```

Making Random Clouds

```
Vector2 position =  
    new Vector2(rand.Next(GraphicsDevice.Viewport.Width),  
                rand.Next(GraphicsDevice.Viewport.Height));  
  
Vector2 speed = new Vector2(rand.Next(0, 100) / 100f, 0);  
  
Cloud c = new Cloud( cloudTexture, position, speed);  
gameSprites.Add(c);
```

- The Cloud class has a constructor that takes a texture, position and speed and creates a Cloud instance
- This is then added to the sprites for this game

Updating Sprites

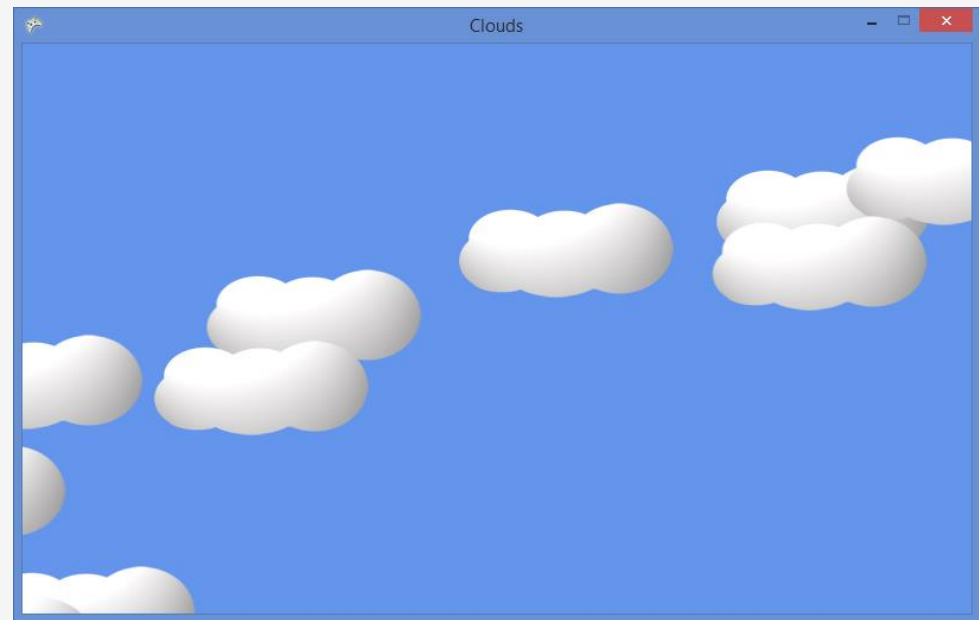
```
protected override void Update(GameTime gameTime)
{
    foreach (ISprite sprite in gameSprites)
        sprite.Update(this);

    base.Update(gameTime);
}
```

- The Update behaviour works through each game component and updates it
- The component is given a reference to the game so that it can affect the game state if required (e.g. update the score)
- There is a similar loop for Draw

3: Lots of Clouds

Rob Miles
University of Hull



demo

Sprite Update Behavior

```
public void Update(CloudGame game)
{
    CloudPosition += CloudSpeed;

    if (CloudPosition.X > game.GraphicsDevice.Viewport.Width)
        CloudPosition.X = -CloudTexture.Width;
}
```

- The sprite has its own Update behaviour
- This moves the cloud and then puts it back on the left when it falls off the screen

Adding a Dark Cloud

- I want the player to be able to hunt the cloud
 - They will control the “dark cloud”
- I want something like a cloud, but with slightly different Draw and Update behaviours
 - The Draw will draw the cloud darker
 - The Update will allow the player to control the cloud
- C# lets us create a new class, based on Cloud but with these behaviours replaced

Dark Cloud Draw

```
public override void Draw(CloudGame game)
{
    game.spriteBatch.Draw(CloudTexture,
        CloudPosition, Color.DarkGray);
}
```

- The replacement Draw method draws the cloud, but uses the colour DarkGrey instead of white
- The cloud is now shown darker on the screen
- You can use this technique to “colour in” draw items

Dark Cloud Update

```
public override void Update(CloudGame game)
{
    Vector2 moveVector = Vector2.Zero;

    if ( game.CurrentKeyboardState.IsKeyDown(Keys.Right) )
        moveVector.X = CloudSpeed.X;

    if ( game.CurrentKeyboardState.IsKeyDown(Keys.Left) )
        moveVector.X = -CloudSpeed.X;

    CloudPosition = CloudPosition + moveVector;
}
```

- The replacement Update method uses the keyboard state to create a vector that allows the player to control the cloud movement

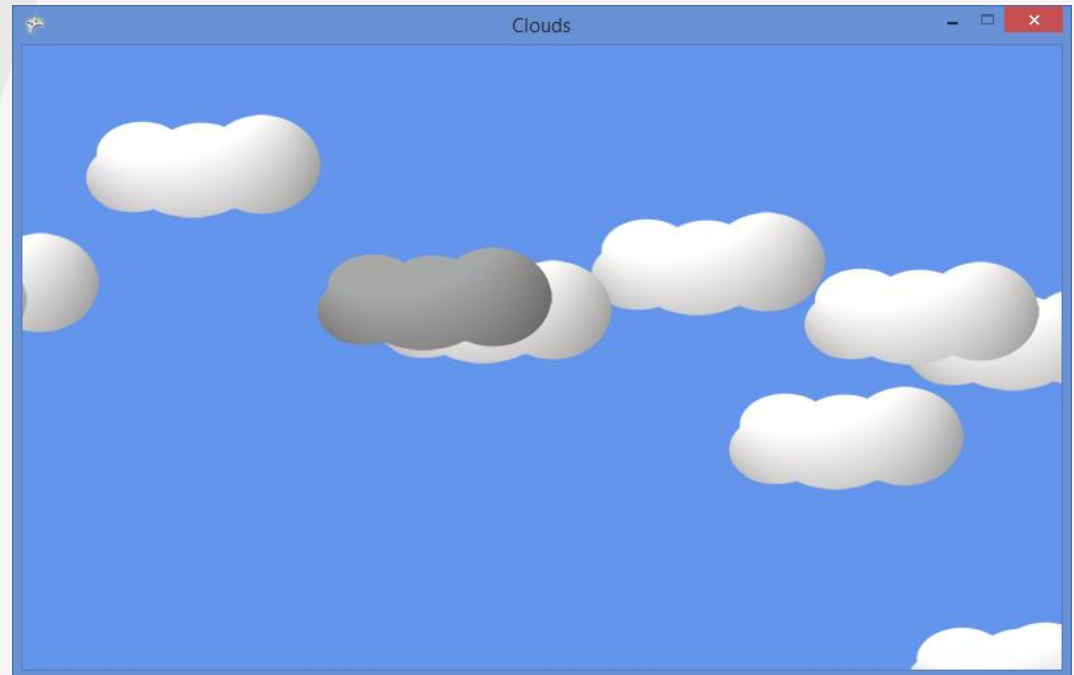
Adding Rectangles

- At the moment the game just draws the texture in the size that it was supplied
- This is not very sensible
- XNA provides a Rectangle class that can be used to position things on the screen
- It only uses integers for position, and so the game must convert the floating point vector values
- You can check for Rectangle intersection

4: Bursting Clouds

Rob Miles
University of Hull

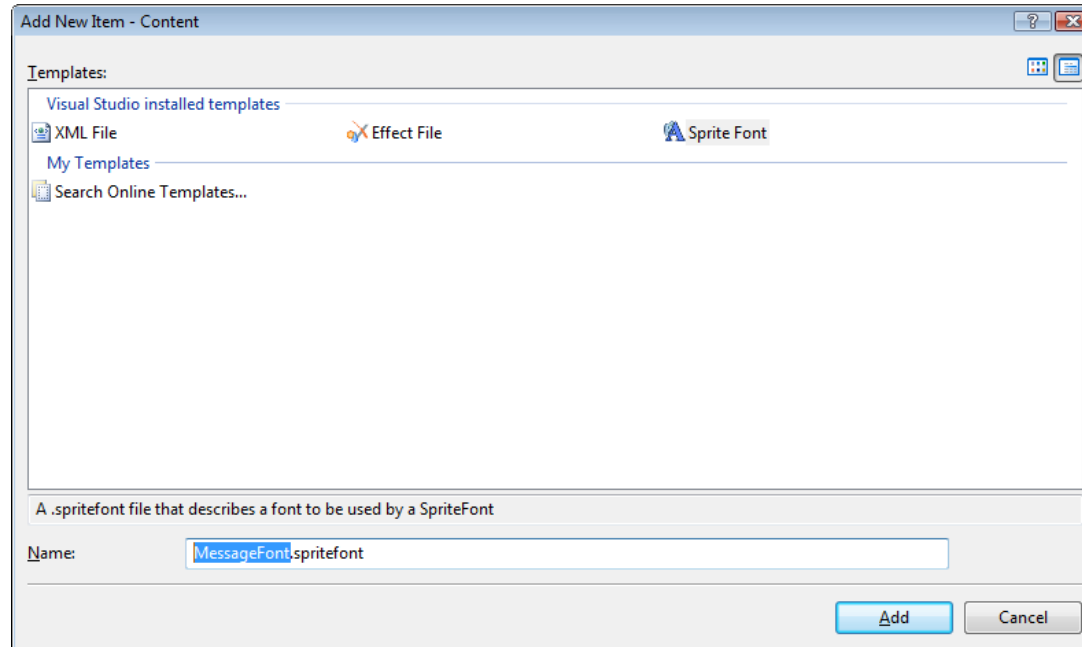
demo



Displaying Text

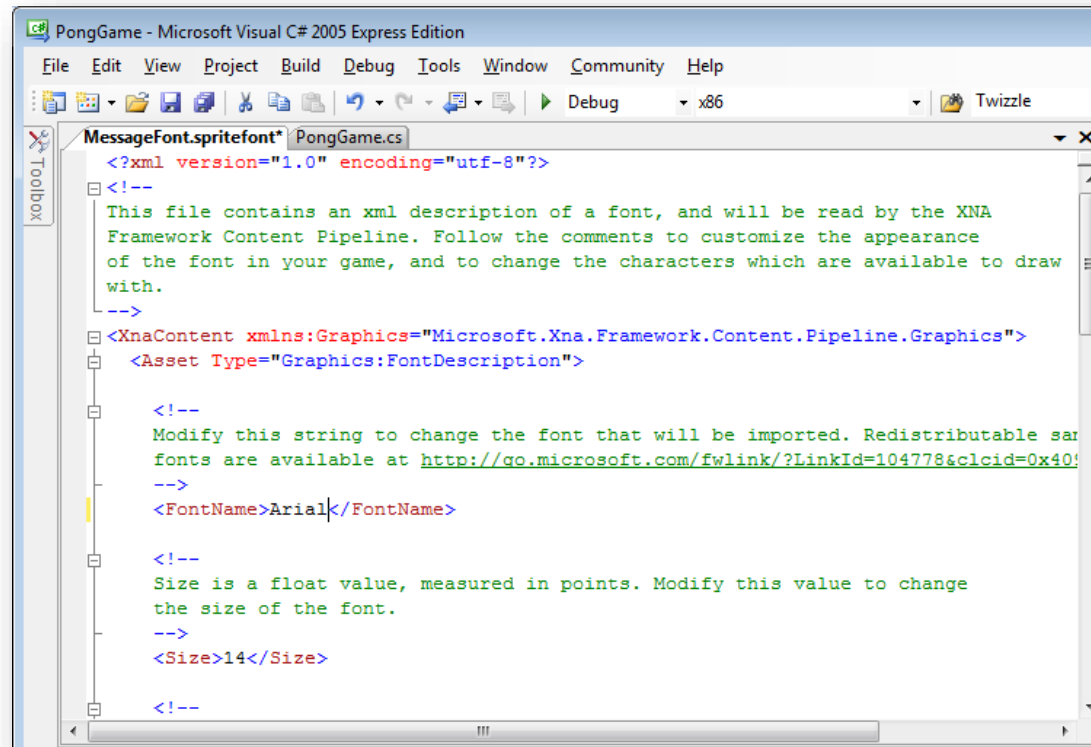
- An XNA game can draw text on the screen
- It does this by rendering a font which contains the character designs
- The font is called a *SpriteFont* and contains a set of textures which are created when the program is built
- This font is loaded into the program when it runs
- It behaves as any other item of content

Adding a SpriteFont



- To add a font:
 - Select the Content Project in Solution Explorer
 - Select Project>Add New Item
 - Select SpriteFont from the dialog

SpriteFont XML



```

PongGame - Microsoft Visual C# 2005 Express Edition
File Edit View Project Build Debug Tools Window Community Help
Debug x86 Twizzle
MessageFont.spritefont* PongGame.cs
<?xml version="1.0" encoding="utf-8"?>
<!--
This file contains an xml description of a font, and will be read by the XNA
Framework Content Pipeline. Follow the comments to customize the appearance
of the font in your game, and to change the characters which are available to draw
with.
-->
<XnaContent xmlns:Graphics="Microsoft.Xna.Framework.Content.Pipeline.Graphics">
  <Asset Type="Graphics:FontDescription">
    <!--
    Modify this string to change the font that will be imported. Redistributable san
    fonts are available at http://go.microsoft.com/fwlink/?LinkId=104778&clcid=0x409c
    -->
    <FontName>Arial</FontName>
    <!--
    Size is a float value, measured in points. Modify this value to change
    the size of the font.
    -->
    <Size>14</Size>
    <!--
  
```

- The font used and the size are set in an XML file
- You can edit this to get different sizes and styles

Loading a Font

```
SpriteFont font;  
  
protected override void LoadContent()  
{  
    // Load the bat and ball textures  
    font = Content.Load<SpriteFont>("MessageFont");  
}
```

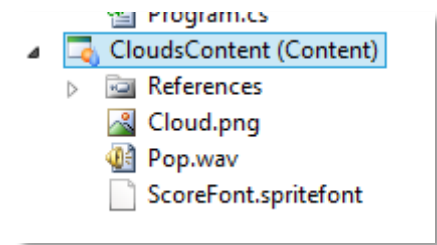
- The Content Manager will fetch the font
- The font can be stored in a variable which is a member of the game class
- You can use multiple fonts if you want different text styles

Sound output

- A sound is just another item of content
- You can use most kinds of files
- I prefer WAV files
- If you are looking for a good audio program I suggest one called Audacity:

<http://audacity.sourceforge.net/>

- It is free and provides lots of useful effects



Using a SoundEffect

```
public SoundEffect PopSound;  
  
//Load the sound in LoadContent  
  
PopSound = Content.Load<SoundEffect>("Pop");  
  
//Play the sound in the game  
  
PopSound.Play();
```

- Calling the Play method on a sound effect causes it to play
- You can have lots of sound effects and play them simultaneously if required

Creating Games

- This set of sprites can be used as the basis of a game
 - Remember that you don't need to have the same texture for every sprite
 - You can draw using different coloured “light”
 - You can make a very big texture that fills the screen, and use that as the background
- Since one sprite can make use of information in another you can make sprites that chase each other, or avoid each other
 - This is the basis of Artificial Intelligence (AI)

5: Completed Game

Rob Miles
University of Hull



demo

The final demo

- The final game also has a Red Cloud that will chase you
- This means that we can make a simple game mechanic where you have to burst all the clouds before the Red Cloud catches you
- This means that the player has to plan a route to the clouds without being caught, so there is a balance between risk and reward
- This is how games are

Getting Going

- Feel free to use the sample games in any way you like
- They are free to download from the Three Thing Game website
- Good luck, and have fun!