

## Introduction

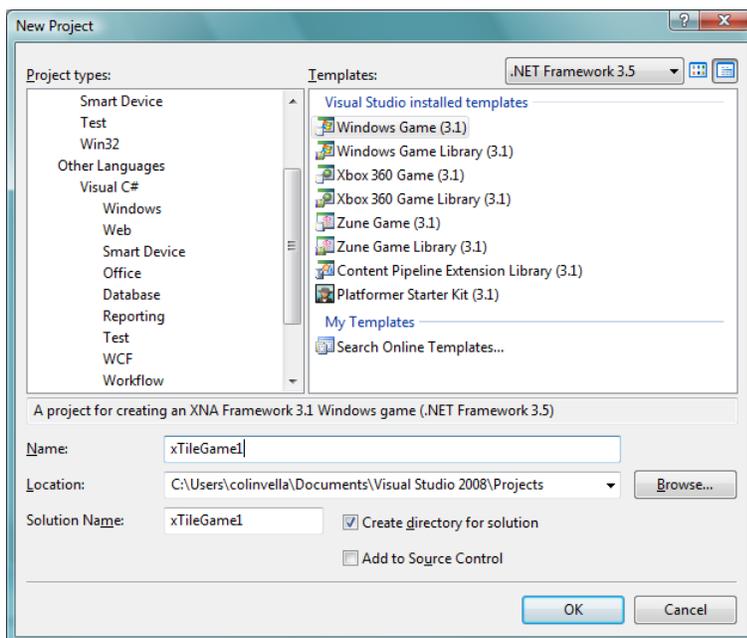
This is a short tutorial explaining how to get started with the **xTile** tiling engine. By the end of this tutorial you should have a working XNA game application for Windows featuring a parallax scrolling animated map.

## Prerequisites

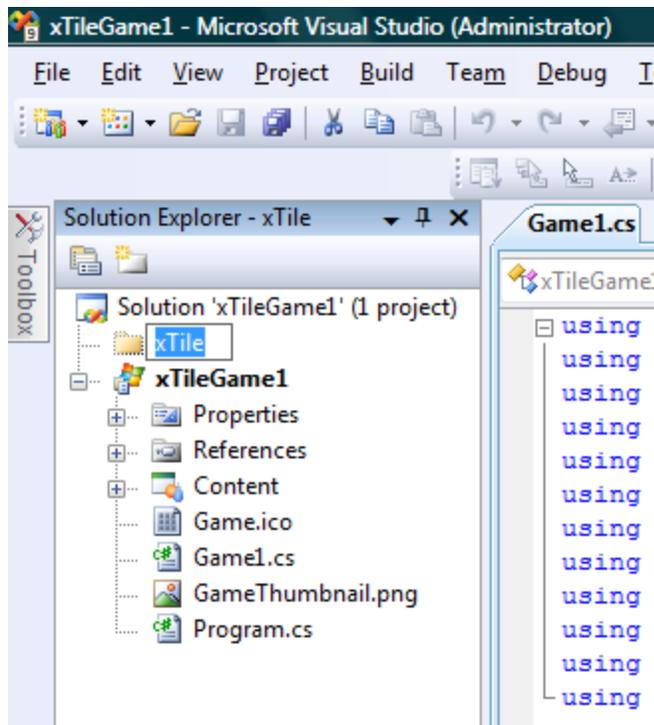
- Visual Studio 2008 (any edition)
- .NET Framework 3.5
- XNA Game Studio 3.1
- xTile Engine libraries (download xTile120.zip or later from <http://tide.codeplex.com/releases>)
- tIDE map content (download Samples111.zip from <http://tide.codeplex.com/releases>)

## Steps

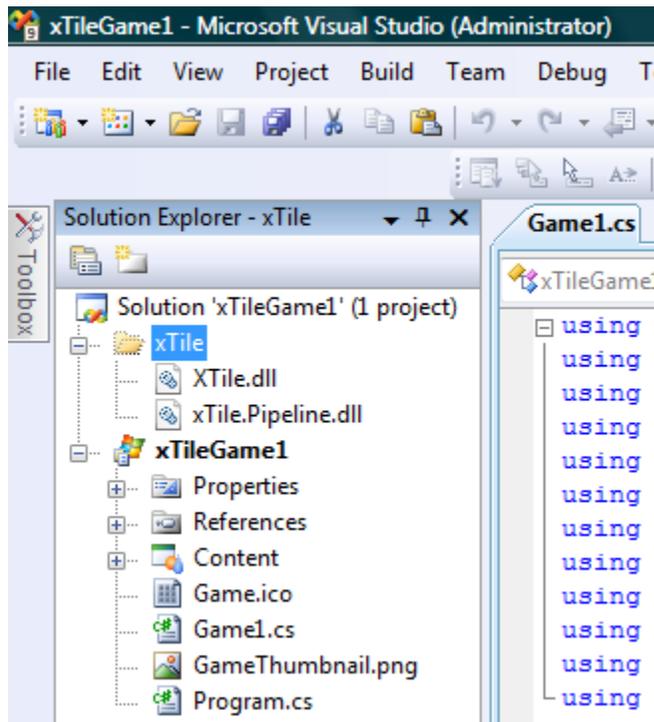
1. Start **Visual Studio 2008** and create a new **XNA Windows Game (3.1)** project and name it **xTileGame1**.



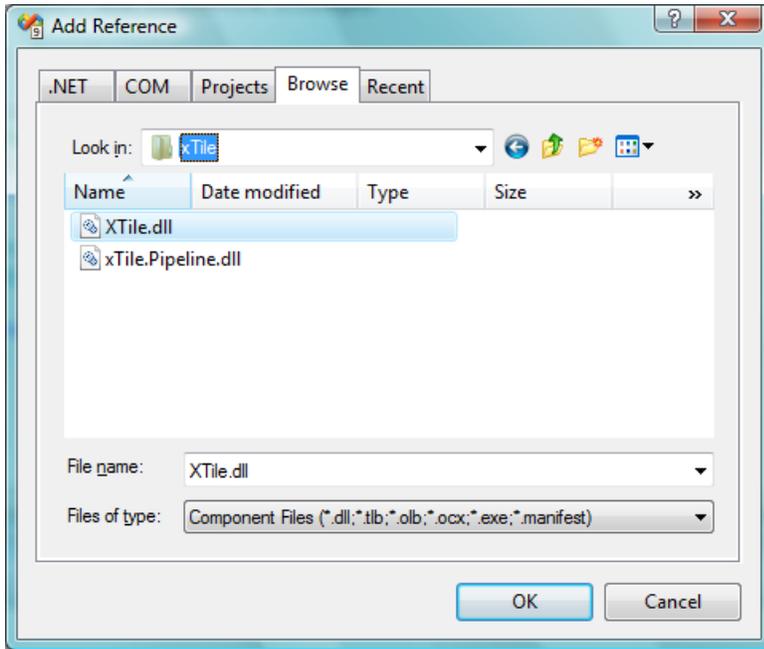
2. Using the **Solution Explorer** panel, create a new solution folder named **xTile** directly under the solution. Also create a corresponding **xTile** folder on disk within the **xTileGame1** solution folder.



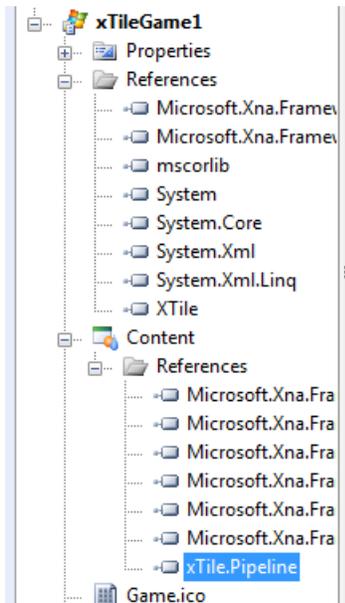
3. Copy the **xTile.dll** (PC version) and **xTile.Pipeline.dll** assemblies from the downloaded **xTile Engine ZIP archive** into the **xTile** solution folder created previously. Add the assemblies as existing items under the **xTile** folder in **Solution Explorer**.



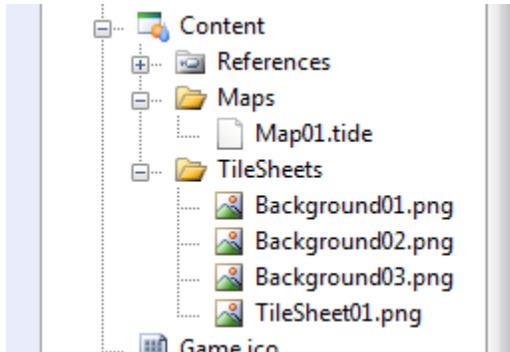
4. Add an **xTile.dll** reference to the **xTileGame1** project.



5. Add an **xTile.Pipeline.dll** reference to the **Content** project of the **xTileGame1** project (*not the game project itself*).



6. Add a **tIDE** map file and its associated tile sheet images in the **Content** project. *Create content folders as necessary to maintain relative path references from the map file to the tile sheet image sources.*



7. Compile the project and verify that the tIDE map file is being correctly processed by the content pipeline.

```

----- Build started: Project: xTileGame1, Configuration: Debug x86 -----
Building Maps\Map01.tide -> C:\Users\colinvella\Documents\Visual Studio 2008\Projects\xTileGame1\xTileGame1\bin\x
Discovered dependency on asset 'TileSheets\Background01.png'...
Converted image source reference '..\TileSheets\Background01.png' to asset reference 'TileSheets\Background01'
Discovered dependency on asset 'TileSheets\Background02.png'...
Converted image source reference '..\TileSheets\Background02.png' to asset reference 'TileSheets\Background02'
Discovered dependency on asset 'TileSheets\Background03.png'...
Converted image source reference '..\TileSheets\Background03.png' to asset reference 'TileSheets\Background03'
Discovered dependency on asset 'TileSheets\TileSheet01.png'...
Converted image source reference '..\TileSheets\TileSheet01.png' to asset reference 'TileSheets\TileSheet01'
Building TileSheets\Background01.png -> C:\Users\colinvella\Documents\Visual Studio 2008\Projects\xTileGame1\xTile
Building TileSheets\Background02.png -> C:\Users\colinvella\Documents\Visual Studio 2008\Projects\xTileGame1\xTile
Building TileSheets\Background03.png -> C:\Users\colinvella\Documents\Visual Studio 2008\Projects\xTileGame1\xTile
Building TileSheets\TileSheet01.png -> C:\Users\colinvella\Documents\Visual Studio 2008\Projects\xTileGame1\xTile
C:\Windows\Microsoft.NET\Framework\v3.5\Csc.exe /noconfig /nowarn:1701,1702 /nostdlib+ /platform:x86 /errorreport
Compile complete -- 0 errors, 0 warnings
xTileGame1 -> C:\Users\colinvella\Documents\Visual Studio 2008\Projects\xTileGame1\xTileGame1\bin\x86\Debug\xTile
===== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped =====

```

8. Access the code file **Game1.cs** and add the following **using** clauses for the **xTile** engine:

```

using Microsoft.Xna.Framework.Net;
using Microsoft.Xna.Framework.Storage;

// xTile engine namespaces
using xTile;
using xTile.Dimensions;
using xTile.Display;

namespace xTileGame1
{

```

9. Add member variables for an **xTile** map, display device and viewport:

```

public class Game1 : Microsoft.Xna.Framework.Game
{
    GraphicsDeviceManager graphics;
    SpriteBatch spriteBatch;

    // xTile map, display device reference, and rendering viewport
    Map map;
    IDisplayDevice mapDisplayDevice;
    xTile.Dimensions.Rectangle viewport;

    public Game1()
    {

```

10. Within the **Initialize()** method, instantiate the map display device, load the map resources and set a display viewport:

```

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

    base.Initialize();

    // initialise xTile map display device
    mapDisplayDevice = new XnaDisplayDevice(
        this.Content, this.GraphicsDevice);

    // initialise xTile map resources
    map.LoadTileSheets(mapDisplayDevice);

    // initialise xTile rendering viewport (hardcoded for now)
    viewport = new xTile.Dimensions.Rectangle(new Size(800, 600));
}

```

11. Within the **LoadContent()** method, load the map from the content pipeline:

```

protected override void LoadContent()
{
    // Create a new SpriteBatch, which can be u
    spriteBatch = new SpriteBatch(GraphicsDevic

    // TODO: use this.Content to load your game

    // load xTile map from content pipeline
    map = Content.Load<Map>("Maps\\Map01");
}

```

12. Within the **Update(GameTime gameTime)** method, update the map for tile animation and update the viewport for level scrolling:

```
protected override void Update(GameTime gameTime)
{
    // Allows the game to exit
    if (GamePad.GetState(PlayerIndex.One).Buttons.Back ==
        this.Exit());

    // TODO: Add your update logic here

    // update xTile map for animations etc.
    // and update viewport for camera movement
    map.Update(gameTime.ElapsedGameTime.Milliseconds);
    viewport.X++;

    base.Update(gameTime);
}
```

13. Within the **Draw(GameTime gameTime)** method, render the map visually using the display device and viewport:

```
protected override void Draw(GameTime gameTime)
{
    GraphicsDevice.Clear(Color.CornflowerBlue);

    // TODO: Add your drawing code here

    // render xTile map
    map.Draw(mapDisplayDevice, viewport);

    base.Draw(gameTime);
}
```

14. Compile and run the **xTileGame1** project.

