

```
1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5
6  using Microsoft.Xna.Framework;
7  using Microsoft.Xna.Framework.Content;
8  using Microsoft.Xna.Framework.Graphics;
9  using Microsoft.Xna.Framework.Input;
10
11 namespace Demo_MG_MazeGame
12 {
13     public class Wall
14     {
15         #region FIELDS
16
17         private ContentManager _contentManager;
18         private string _spriteName;
19         private Texture2D _sprite;
20         private Vector2 _position;
21         private Vector2 _center;
22         private Rectangle _boundingRectangle;
23
24         private bool _active;
25
26         #endregion
27
28         #region PROPERTIES
29
30         public ContentManager ContentManager
31         {
32             get { return _contentManager; }
33             set { _contentManager = value; }
34         }
35
36         public string SpriteName
37         {
38             get { return _spriteName; }
39             set { _spriteName = value; }
40         }
41
42         public Vector2 Position
43         {
44             get { return _position; }
45             set
46             {
47                 _position = value;
48                 _center = new Vector2(_position.X + (_sprite.Width / 2),
49                                     _position.Y + (_sprite.Height / 2));
49                 _boundingRectangle = new Rectangle((int)_position.X, (int)
50                                                     _position.Y, _sprite.Width, _sprite.Height);
51             }
52         }
53
54         public Vector2 Center
55         {
56             get { return _center; }
57             set { _center = value; }
58         }
59
60         public Rectangle BoundingRectangle
61         {
62             get { return _boundingRectangle; }
63             set { _boundingRectangle = value; }
64         }
65
66         public bool Active
67         {
68             get { return _active; }
69             set { _active = value; }
70         }
71     }
72 }
```

```
67         get { return _active; }
68         set { _active = value; }
69     }
70
71     #endregion
72
73     #region CONSTRUCTORS
74
75     /// <summary>
76     /// instantiate a new Wall
77     /// </summary>
78     /// <param name="contentManager">game content manager object</param>
79     /// <param name="spriteName">file name of sprite</param>
80     /// <param name="position">vector position of Wall</param>
81     public Wall(
82         ContentManager contentManager,
83         string spriteName,
84         Vector2 position
85     )
86     {
87         _contentManager = contentManager;
88         _spriteName = spriteName;
89         _position = position;
90
91         // load the Wall image into the Texture2D for the Wall sprite
92         _sprite = _contentManager.Load<Texture2D>(_spriteName);
93
94         // set the initial center and bounding rectangle for the wall
95         _center = new Vector2(position.X + (_sprite.Width / 2), position.Y + (
96             _sprite.Height / 2));
97         _boundingRectangle = new Rectangle((int)position.X, (int)position.Y,
98             _sprite.Width, _sprite.Height);
99     }
100
101     #endregion
102
103     #region METHODS
104     /// <summary>
105     /// add Wall sprite to the SpriteBatch object
106     /// </summary>
107     /// <param name="spriteBatch"></param>
108     public void Draw(SpriteBatch spriteBatch)
109     {
110         // only draw the Wall if it is active
111         if (_active)
112         {
113             spriteBatch.Draw(_sprite, _position, Color.White);
114         }
115     }
116
117     #endregion
118 }
```