```
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 Player
14     {
15         #region ENUMS
16
17         public enum Direction
18         {
19             Left,
20             Right
21         }
22
23         #endregion
24
25         #region FIELDS
26
27         private ContentManager _contentManager;
28         private int _spriteWidth;
29         private int _spriteHeight;
30         private Texture2D _sprite;
31         private Vector2 _position;
32         private Vector2 _center;
33         private int _speedHorizontal;
34         private int _speedVertical;
35         private Direction _DirectionOfTravel;
36         private Rectangle _boundingRectangle;
37         private bool _active;
38
39         #endregion
40
41         #region PROPERTIES
42
43         public ContentManager ContentManager
44         {
45             get { return _contentManager; }
46             set { _contentManager = value; }
47         }
48
49         public Vector2 Position
50         {
51             get { return _position; }
52             set
53             {
54                 _position = value;
55                 _center = new Vector2(_position.X + (_spriteWidth / 2), _position
                   .Y + (_spriteHeight / 2));
56                 _boundingRectangle = new Rectangle((int)_position.X, (int)
                   _position.Y, _spriteWidth, _spriteHeight);
57             }
58         }
59
60         public Vector2 Center
61         {
62             get { return _center; }
63             set { _center = value; }
64         }
65
66         public int SpeedHorizontal
```

```
67              {
68                  get { return _speedHorizontal; }
69                  set { _speedHorizontal = value; }
70              }
71
72              public int SpeedVertical
73              {
74                  get { return _speedVertical; }
75                  set { _speedVertical = value; }
76              }
77
78              public Direction PlayerDirection
79              {
80                  get { return _DirectionOfTravel; }
81                  set { _DirectionOfTravel = value; }
82              }
83
84              public Rectangle BoundingRectangle
85              {
86                  get { return _boundingRectangle; }
87                  set { _boundingRectangle = value; }
88              }
89
90              public bool Active
91              {
92                  get { return _active; }
93                  set { _active = value; }
94              }
95
96              #endregion
97
98              #region CONSTRUCTORS
99
100             /// <summary>
101             /// instantiate a new Player
102             /// </summary>
103             /// <param name="contentManager">game content manager object</param>
104             /// <param name="spriteName">file name of sprite</param>
105             /// <param name="position">vector position of Player</param>
106             public Player(
107                 ContentManager contentManager,
108                 Vector2 position
109                 )
110             {
111                 _contentManager = contentManager;
112                 _position = position;
113
114                 // load the Player image
115                 _sprite = _contentManager.Load<Texture2D>("player");
116
117                 _spriteWidth = _sprite.Width;
118                 _spriteHeight = _sprite.Height;
119
120                 // set the initial center and bounding rectangle for the Player
121                 _center = new Vector2(position.X + (_spriteWidth / 2), position.Y + (
                    _spriteHeight / 2));
122                 _boundingRectangle = new Rectangle((int)position.X, (int)position.Y,
                    _spriteWidth, _spriteHeight);
123             }
124
125             #endregion
126
127             #region METHODS
128
129             /// <summary>
130             /// add Player sprite to the SpriteBatch object
131             /// </summary>
132             /// <param name="spriteBatch"></param>
```

```
133             public void Draw(SpriteBatch spriteBatch)
134             {
135                 // only draw the Player if it is active
136                 if (_active)
137                 {
138                         spriteBatch.Draw(_sprite, _position, Color.White);
139                 }
140             }
141
142             #endregion
143         }
144     }
145
```