

```
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_PlatformMovement
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 Texture2D _spriteRight;
29         private Texture2D _spriteLeft;
30         private int _spriteWidth;
31         private int _spriteHeight;
32         private Vector2 _position;
33         private Vector2 _center;
34         private int _speedHorizontal;
35         private int _speedVertical;
36         private Direction _DirectionOfTravel;
37         private Rectangle _boundingRectangle;
38         private bool _active;
39
40         #endregion
41
42         #region PROPERTIES
43
44         public ContentManager ContentManager
45         {
46             get { return _contentManager; }
47             set { _contentManager = value; }
48         }
49
50         public Vector2 Position
51         {
52             get { return _position; }
53             set
54             {
55                 _position = value;
56                 _center = new Vector2(_position.X + (_spriteWidth / 2), _position
57                                     .Y + (_spriteHeight / 2));
58                 _boundingRectangle = new Rectangle((int)_position.X, (int)
59                                     _position.Y, _spriteWidth, _spriteHeight);
60             }
61         }
62
63         public Vector2 Center
64         {
65             get { return _center; }
66             set { _center = value; }
```

```

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

```

```

131 | /// <summary>
132 | /// add Player sprite to the SpriteBatch object
133 | /// </summary>
134 | /// <param name="spriteBatch"></param>
135 | public void Draw(SpriteBatch spriteBatch)
136 | {
137 |     // only draw the Player if it is active
138 |     if (_active)
139 |     {
140 |         if (_DirectionOfTravel == Direction.Right)
141 |         {
142 |             spriteBatch.Draw(_spriteRight, _position, Color.White);
143 |         }
144 |         else
145 |         {
146 |             spriteBatch.Draw(_spriteLeft, _position, Color.White);
147 |         }
148 |     }
149 | }
150 |
151 | #endregion
152 | }
153 | }
154 |

```