```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Content;
using Microsoft.Xna.Framework.Graphics;
using Microsoft.Xna.Framework.Input;

namespace Demo_MG_PlatformMovement
{
    public class Player
    {
        #region ENUMS

        public enum Direction
        {
            Left,
            Right
        }

        #endregion

        #region FIELDS

        private ContentManager _contentManager;
        private Texture2D _spriteRight;
        private Texture2D _spriteLeft;
        private int _spriteWidth;
        private int _spriteHeight;
        private Vector2 _position;
        private Vector2 _center;
        private int _speedHorizontal;
        private int _speedVertical;
        private Direction _DirectionOfTravel;
        private Rectangle _boundingRectangle;
        private bool _active;

        #endregion

        #region PROPERTIES

        public ContentManager ContentManager
        {
            get { return _contentManager; }
            set { _contentManager = value; }
        }

        public Vector2 Position
        {
            get { return _position; }
            set
            {
                _position = value;
                _center = new Vector2(_position.X + (_spriteWidth / 2), _position.Y + (_spriteHeight / 2));
                _boundingRectangle = new Rectangle((int)_position.X, (int)_position.Y, _spriteWidth, _spriteHeight);
            }
        }

        public Vector2 Center
        {
            get { return _center; }
            set { _center = value; }
        }

        public int SpeedHorizontal
        {
            get { return _speedHorizontal; }
            set { _speedHorizontal = value; }
        }

        public int SpeedVertical
        {
            get { return _speedVertical; }
            set { _speedVertical = value; }
        }

        public Direction PlayerDirection
        {
            get { return _DirectionOfTravel; }
            set { _DirectionOfTravel = value; }
        }

        public Rectangle BoundingRectangle
        {
            get { return _boundingRectangle; }
            set { _boundingRectangle = value; }
        }

        public bool Active
        {
            get { return _active; }
            set { _active = value; }
        }

        #endregion

        #region CONSTRUCTORS

        /// <summary>
        /// instantiate a new Player
        /// </summary>
        /// <param name="contentManager">game content manager object</param>
        /// <param name="spriteName">file name of sprite</param>
        /// <param name="position">vector position of Player</param>
        public Player(
            ContentManager contentManager,
            Vector2 position
            )
        {
            _contentManager = contentManager;
            _position = position;

            // load the Player images in for the different directions of travel
            _spriteLeft = _contentManager.Load<Texture2D>("player_left");
            _spriteRight = _contentManager.Load<Texture2D>("player_right");

            _spriteWidth = _spriteLeft.Width;
            _spriteHeight = _spriteLeft.Height;

            // set the initial center and bounding rectangle for the Player
            _center = new Vector2(position.X + (_spriteWidth / 2), position.Y + (_spriteHeight / 2));
            _boundingRectangle = new Rectangle((int)position.X, (int)position.Y, _spriteWidth, _spriteHeight);
        }

        #endregion

        #region METHODS

        /// <summary>
        /// add Player sprite to the SpriteBatch object
        /// </summary>
        /// <param name="spriteBatch"></param>
        public void Draw(SpriteBatch spriteBatch)
        {
            // only draw the Player if it is active
            if (_active)
            {
                if (_DirectionOfTravel == Direction.Right)
                {
                    spriteBatch.Draw(_spriteRight, _position, Color.White);

                }
                else
                {
                    spriteBatch.Draw(_spriteLeft, _position, Color.White);
                }
            }
        }

        #endregion
    }
}
```