```
 1  using CustomProject.GameHandler;
 2  using CustomProject.GameObjects;
 3  using CustomProject.Observers;
 4  using CustomProject.StrategyDesign;
 5  using SplashKitSDK;
 6
 7  namespace CustomProject
 8  {
 9      public class Player : DrawableObject, IHaveInventory, ISubject
10      {
11          private double _gravity;
12          private AnimationScript _playerScript;
13          private Sprite _playerSprite;
14          private GameMap _gameMap;
15          private bool _isDead;
16          private bool _isMoving;
17
18          private Inventory _inventory;
19          private int _collectedGolds;
20          private int _collectedSwords;
21          private bool _increaseSpeed;
22          private bool _increaseJump;
23
24          private IMovementStrategy _movementStrategy;
25          private IJumpStrategy _jumpStrategy;
26          private ICollectStrategy _collectStrategy;
27
28          private List<IObserver> _observers;
29
30          private AnimationHandler _animationHandler;
31          private CameraHandler _cameraHandler;
32          private CollisionHandler _collisionHandler;
33
34          public Player(GameMap gameMap, string name, string description) :
              base(200, 200, new string[] { "inventory" }, name, description,
              "playerImage.png")
35          {
36              InitializePlayerSprite();
37              InitializePlayerState(gameMap);
38              InitializeStrategy();
39
40              _animationHandler = new AnimationHandler(this);
41              _cameraHandler = new CameraHandler(gameMap, this);
42              _collisionHandler = new CollisionHandler(gameMap, this);
43
44              _observers = new List<IObserver>();
45          }
46
47          // Update the actions of the player
```

```
48          public void Update()
49          {
50              if (_isDead)
51              {
52                  _animationHandler.UpdateDeadAnimation();
53              }
54
55              _movementStrategy.Move(this);
56              _collectStrategy.Collect(this);
57              UpdateYLocation();
58
59              _animationHandler.UpdateMovingAnimation();
60              _cameraHandler.HandleCamera();
61              _playerSprite.UpdateAnimation();
62              _collisionHandler.CheckBombCollision();
63          }
64
65          private void UpdateYLocation()
66          {
67              if (!_isDead) // Check for jumping keypressed
68              {
69                  _jumpStrategy.Jump(this);
70              }
71
72              // Update the gravity and Y location
73              _gravity += 1;
74              double newY = _yLocation + _gravity;
75
76              if (_gravity > 0 && _collisionHandler.IsCollideWithLand
                  (_xLocation, newY))
77              {
78                  _gravity = 0;
79              }
80              else if (_gravity < 0 && _collisionHandler.IsCollideWithLand
                  (_xLocation, newY))
81              {
82                  _gravity = 0;
83              }
84              else
85              {
86                  _yLocation = newY;
87              }
88          }
89
90          // Draw the player on the screen
91          public override void Draw()
92          {
93              _playerSprite.Draw(_xLocation, _yLocation);
94          }
```

```
 95
 96          private void InitializePlayerSprite()
 97          {
 98              _image.SetCellDetails(_image.Width / 8, _image.Height / 6, 8,
                   6, 48);
 99              _playerScript = SplashKit.LoadAnimationScript("_playerScript",
                    "animationscript.txt");
100              _playerSprite = SplashKit.CreateSprite(_image, _playerScript);
101              _playerSprite.StartAnimation(0);
102          }
103
104          private void InitializePlayerState(GameMap gameMap)
105          {
106              _gravity = 0;
107              _isDead = false;
108              _isMoving = false;
109              _gameMap = gameMap;
110              _inventory = new Inventory();
111              _collectedGolds = 0;
112              _collectedSwords = 0;
113              _increaseSpeed = false;
114              _increaseJump = false;
115          }
116
117          private void InitializeStrategy()
118          {
119              _movementStrategy = new NormalMovement();
120              _jumpStrategy = new NormalJump();
121              _collectStrategy = new NormalCollect();
122          }
123
124          public void Attach(IObserver observer)
125          {
126              _observers.Add(observer);
127          }
128
129          public void Detach(IObserver observer)
130          {
131              _observers.Remove(observer);
132          }
133
134          public void Notify(string eventType)
135          {
136              foreach (IObserver observer in _observers)
137              {
138                  observer.Update(eventType, this);
139              }
140          }
141
```

```
142         public GameObject Locate(string id)
143         {
144             if (AreYou(id))
145             {
146                 return this;
147             }
148             else if (_inventory.HasItem(id))
149             {
150                 return _inventory.Fetch(id);
151             }
152             else
153             {
154                 return null;
155             }
156         }
157 
158         public void SetMovementStrategy(IMovementStrategy strategy)
159         {
160             _movementStrategy = strategy;
161         }
162 
163         public void SetJumpStrategy(IJumpStrategy strategy)
164         {
165             _jumpStrategy = strategy;
166         }
167 
168         public void SetCollectStrategy(ICollectStrategy strategy)
169         {
170             _collectStrategy = strategy;
171         }
172 
173         public override string FullDescription
174         {
175             get
176             {
177                 string fulldesc = "\n" + "-----INVENTORY-----" + "\n";
178                 fulldesc += $"You are {Name}, {base.FullDescription}\n";
179                 fulldesc += "You are carrying\n";
180                 fulldesc += $"{_inventory.ItemList}";
181                 return fulldesc;
182             }
183         }
184 
185         public Inventory Inventory
186         {
187             get => _inventory;
188             set => _inventory = value;
189         }
190 
```

```
191         public double Gravity { get => _gravity; set => _gravity =
              value; }
192         public bool IsMoving { get => _isMoving; set => _isMoving =
              value; }
193         public bool IsDead { get => _isDead; set => _isDead = value; }
194         public GameMap GameMap { get => _gameMap; }
195         public int CollectedGolds { get => _collectedGolds; set =>
              _collectedGolds = value; }
196         public int CollectedSwords { get => _collectedSwords; set =>
              _collectedSwords = value; }
197         public bool IncreaseSpeed { get => _increaseSpeed; set =>
              _increaseSpeed = value; }
198         public bool IncreaseJump { get => _increaseJump; set =>
              _increaseJump = value; }
199         public Sprite PlayerSprite { get => _playerSprite; }
200         public CollisionHandler CollisionHandler { get =>
              _collisionHandler; }
201     }
202 }
203
204
```