SolarSoft Platformer Free

A Lightweight, Beginner-Friendly 2D Platformer Framework for Unity

By SolarSoft Media, LLC.

support@solarsoftmedia.com

Version 1.0

March 23, 2025

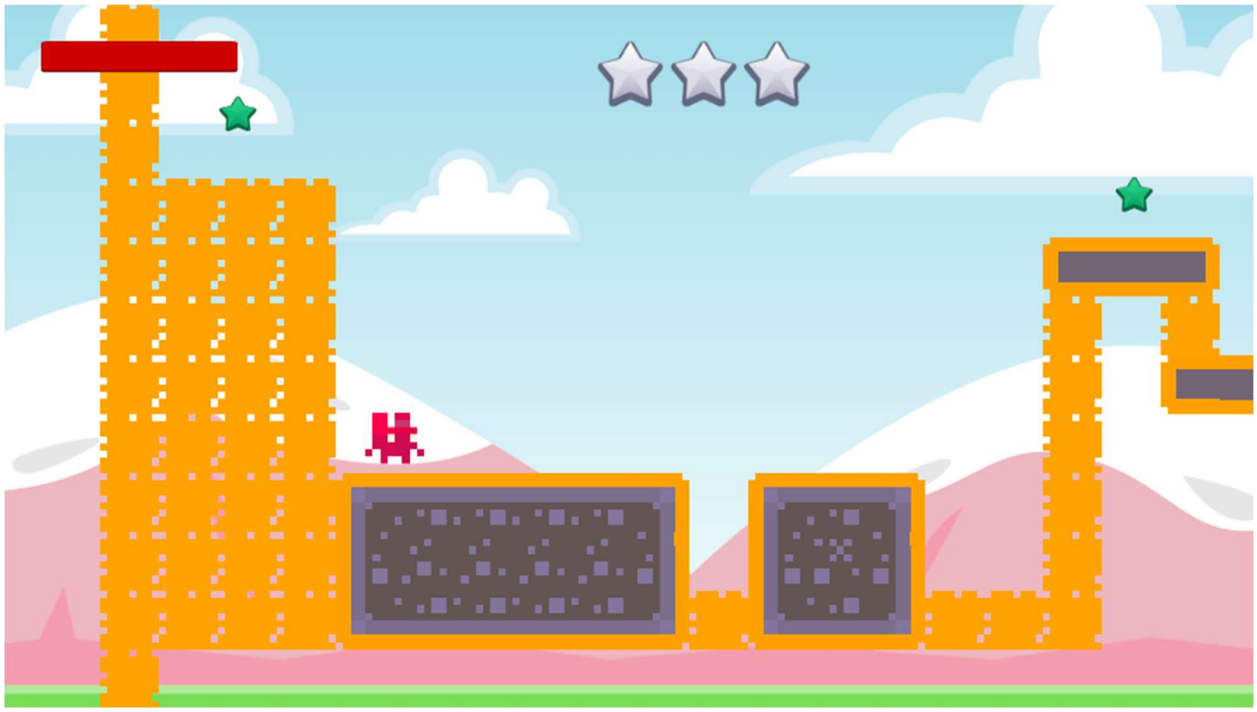


Table of Contents

Introduction

Welcome to SolarSoft Platformer Free – a lightweight, beginner-friendly 2D platformer framework built in Unity. This framework was created to help new and intermediate developers learn core platforming mechanics in a clean, modular, and extensible way.

Whether you're an educator, solo developer, or aspiring Unity programmer, *SolarSoft Platformer Free* provides a solid foundation for building your own side-scrolling platformers, teaching game design, or creating prototypes quickly. Every script is thoughtfully commented to help you understand how and why it works.

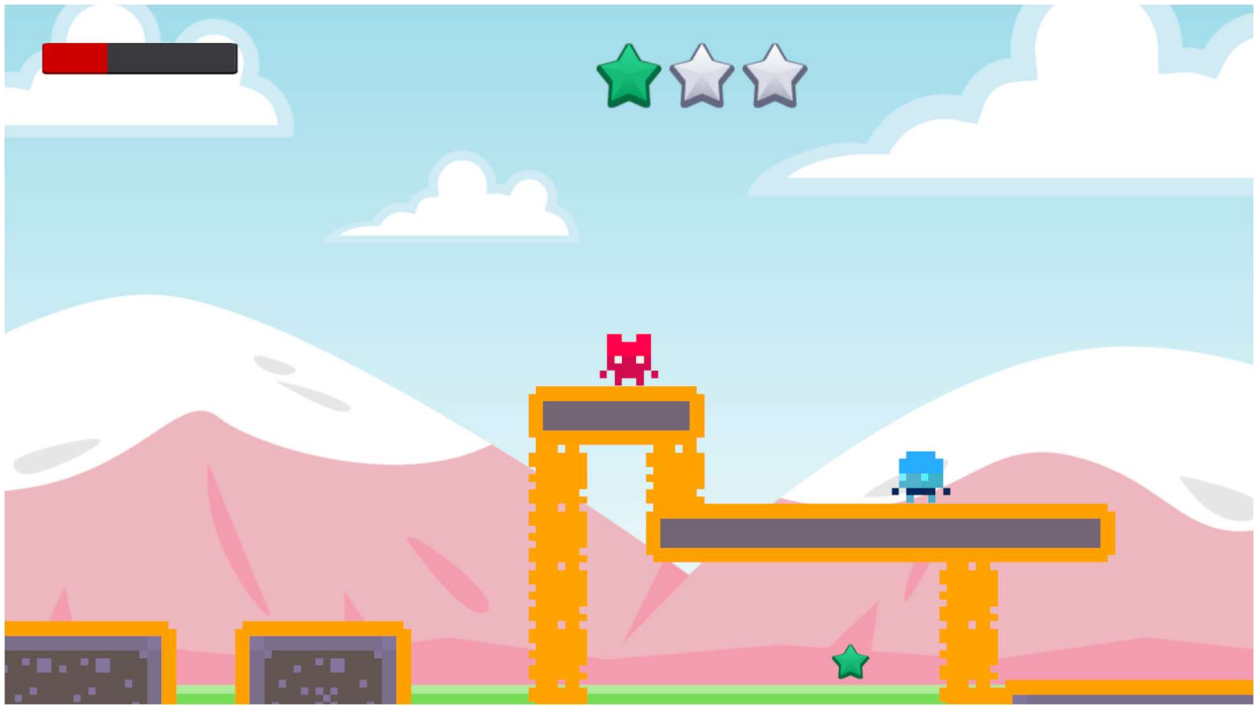
The included demo level showcases all core features, using free and open assets from **Kenney.nl**, so you can jump right into exploring, editing, or expanding the system.

What's Included

This framework includes:

- Smooth 2D player movement (run, jump, coyote time, variable jump height)
- Basic enemy AI with patrol and collision damage
- Collectible stars with a win condition
- Checkpoint and respawn system
- Health system with damage on touch
- Win panel with UI buttons (Restart/Exit)
- Demo scene with modular components
- Clean, well-commented C# scripts

All of this is designed to work out-of-the-box with minimal setup.



Getting Started

This section will guide you through importing, running, and exploring the demo scene included in **SolarSoft Platformer Free**. Setup is minimal, and you'll be up and running in just a few steps.

Importing the Package

To use the framework:

- Open your Unity project or create a new one (2D URP or 2D Core Template recommended).
- Import the **SolarSoft Platformer Free** .unitypackage file.
- The contents will appear under:

  Assets/SolarSoft_Platformer_Free/

Optional: Create a clean folder structure in your project by placing the framework into its own subfolder (e.g., ThirdParty or Frameworks).

Playing the Demo

To explore the sample level:

- Navigate to:

  Assets/SolarSoft_Platformer_Free/Scenes/Demo.unity

- Double-click to open the scene.
- Press **Play** in the Unity Editor.

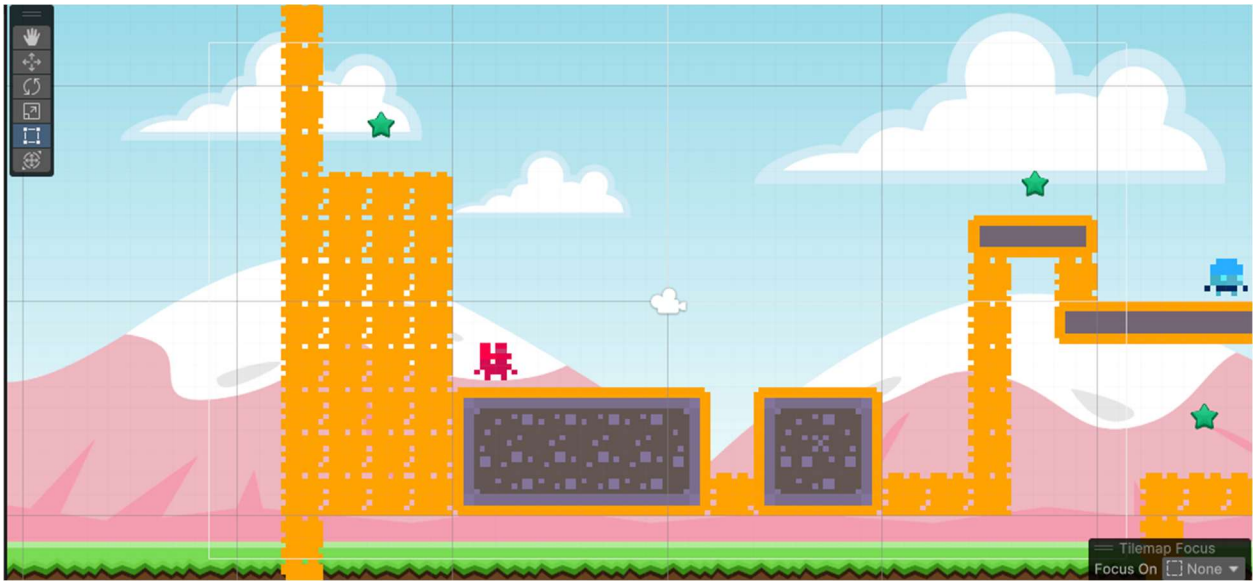
Controls:

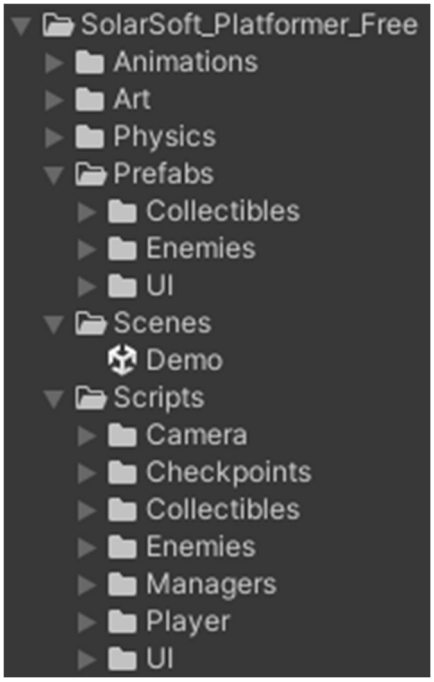
- ← / → or A / D to move
- Space to jump
- Touch an enemy to take damage
- Collect all **3 stars** to win the level

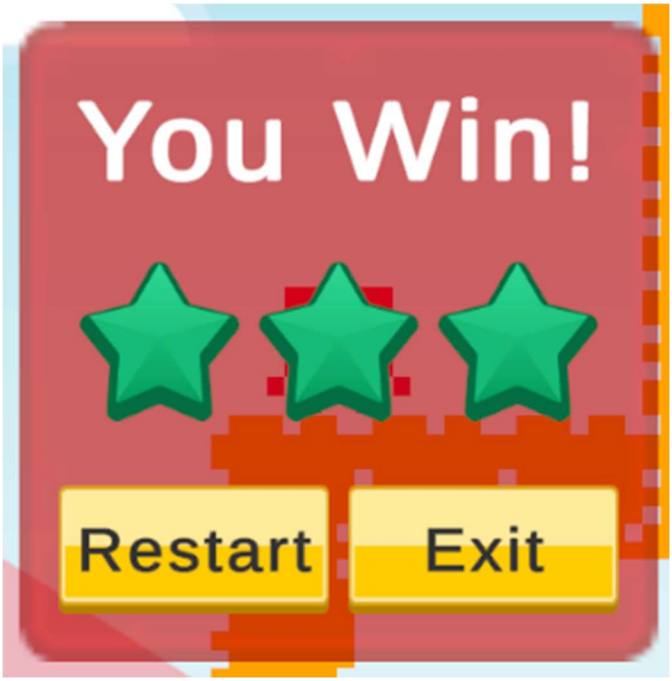
Understanding the Structure

The project is modular and organized for learning:

Folder	Contents
Player	Player controller, health, animations
Enemies	Patrol logic, touch damage system
UI	Health bar, WinPanel, star counter
Collectibles	Star logic, collection tracking
Checkpoints	Respawn and checkpoint manager
Scenes	Demo scene







Feature Overview

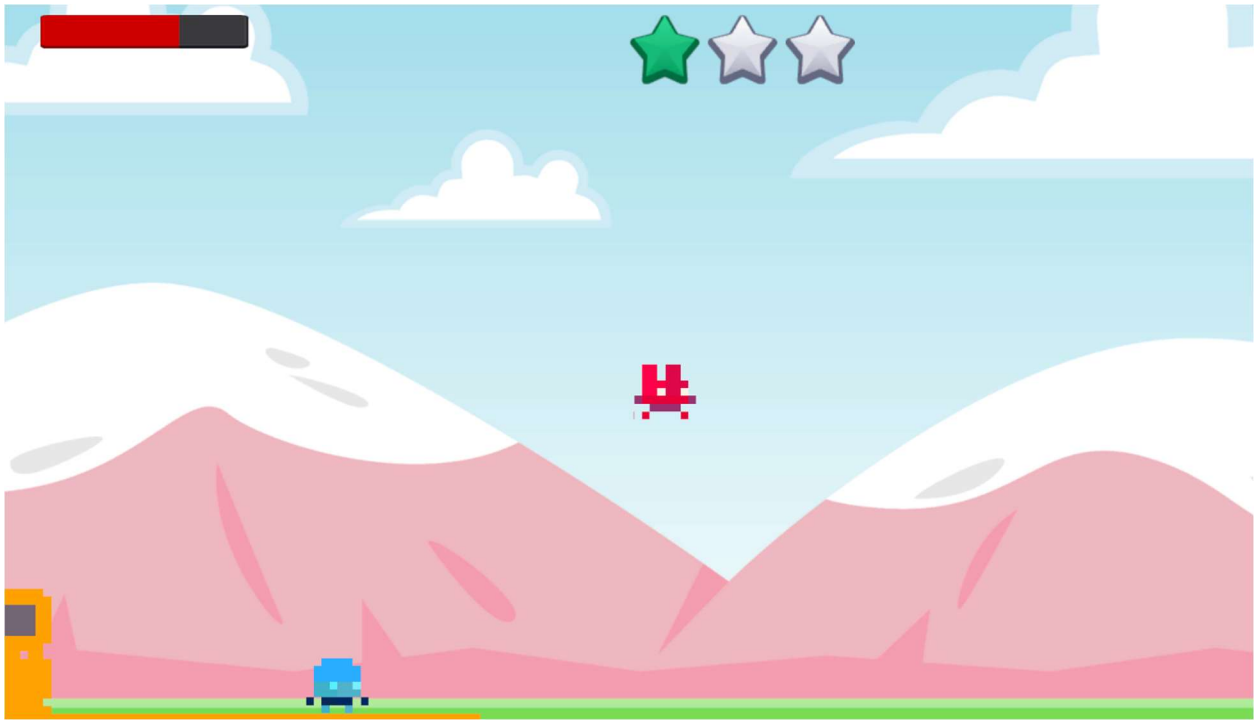
SolarSoft Platformer Free comes with thoughtfully designed systems built to demonstrate core 2D platformer mechanics. Each feature is implemented with clean, modular code to help you learn and extend easily.

Core Player Controller

Features:

- Smooth horizontal movement (A/D or ←/→)
- Jumping with adjustable force and gravity
- **Coyote Time** – Allows jumping shortly after leaving a platform
- **Variable Jump Height** – Tap vs. hold to change jump strength

Editable via the PlayerMovement script.

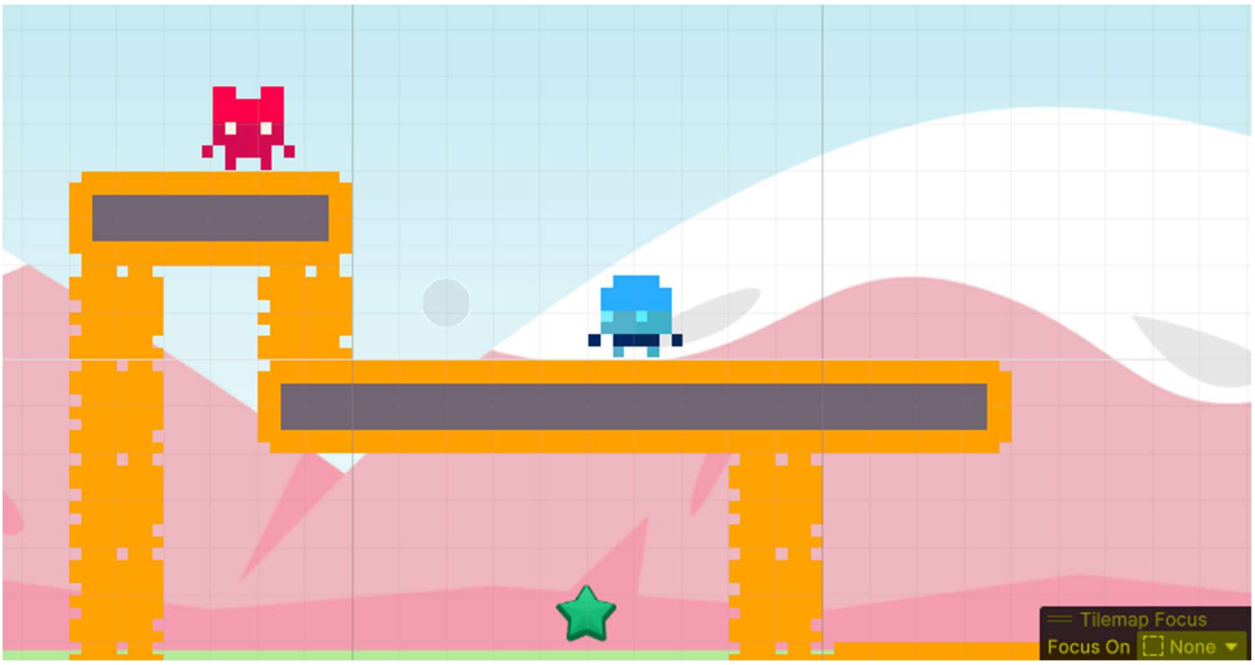


Simple Enemy AI

Features:

- Patrols between two points using EnemyPatrol
- Flips sprite direction when changing direction
- Damages player on contact using TouchDamage

Customize patrol points in the Inspector.

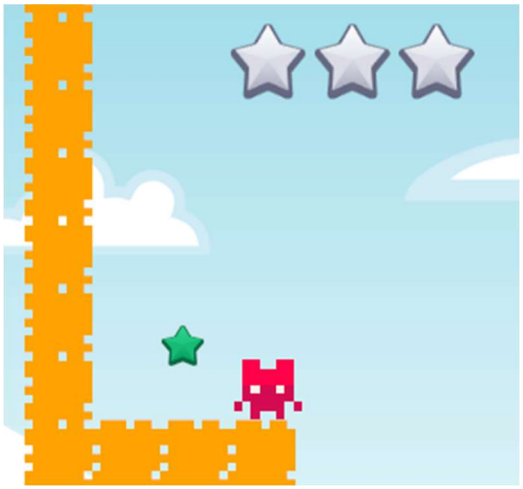


Collectibles System

Features:

- Collectible stars (CollectibleStar)
- UI tracker updates on collection
- Triggers win condition after 3 stars

Add more stars easily in the demo scene.

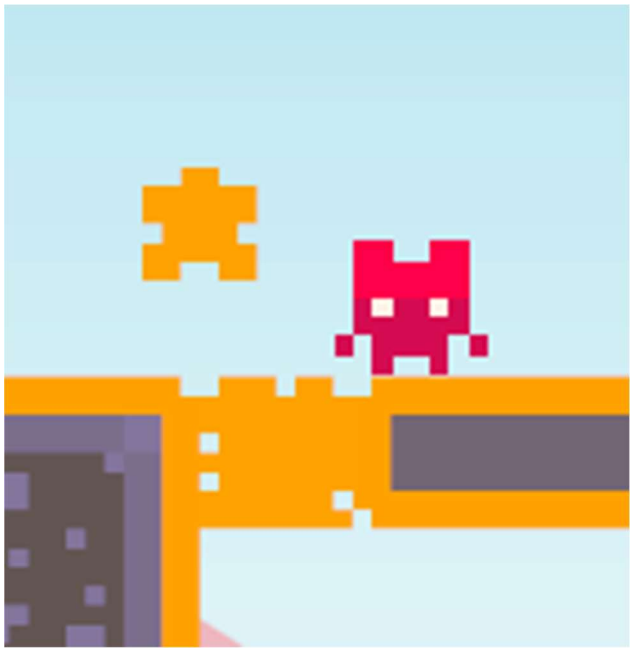


Checkpoint & Respawn System

Features:

- Touching a checkpoint saves your position
- Respawn after death to last checkpoint (or start if none)
- Handled with a CheckpointManager singleton

Can be reused for level progression or autosaves.



This puzzle piece is the checkpoint for demo purposes.

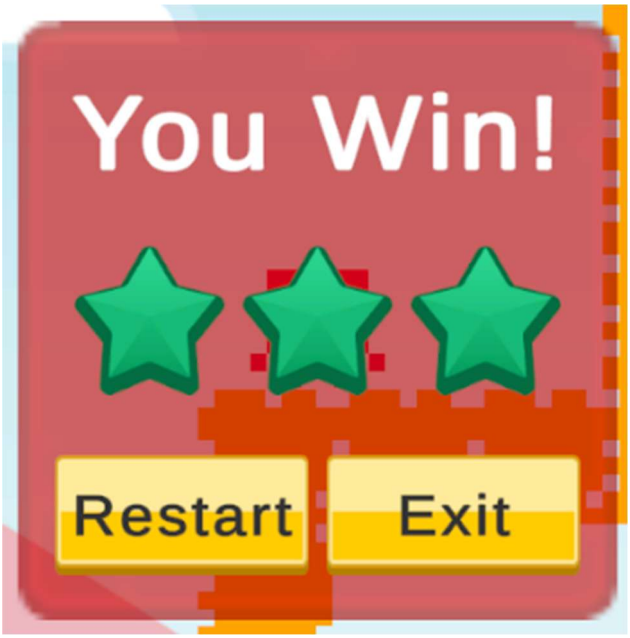
Feel free to change this and any assets to your preference.

UI Elements

Includes:

- Health bar with smooth fill transitions
- Win Panel with restart and exit buttons
- Star UI counter using Image[ ]

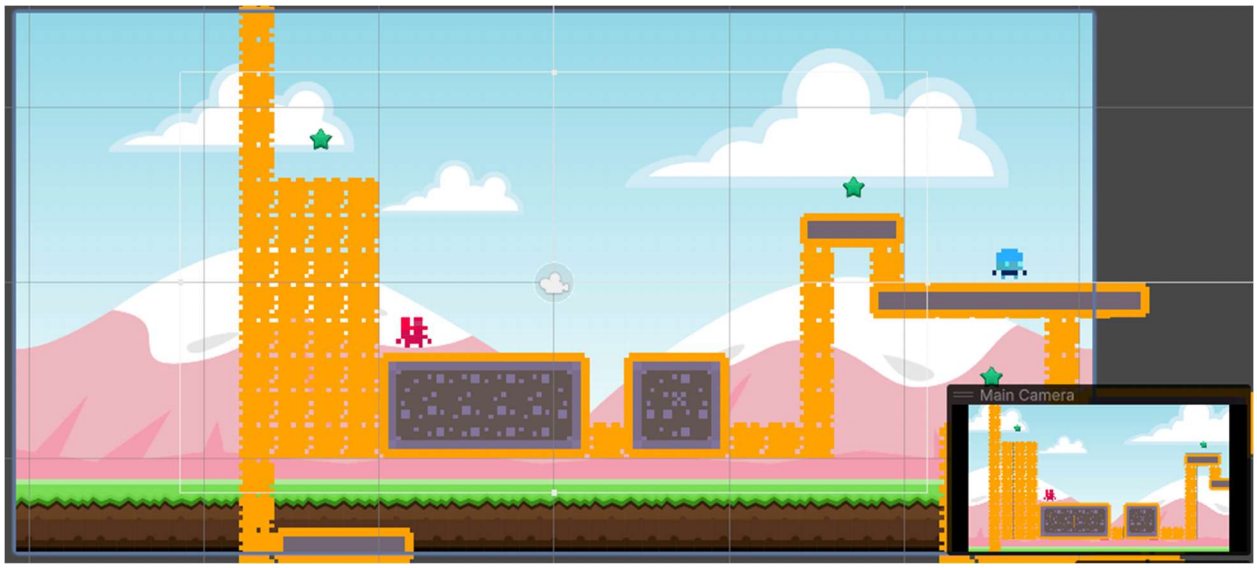
Ui hooks are exposed in the Inspector for easy customization.



Camera Follow

Features:

- Smooth follow with damping
- Optional offset settings
- Controlled via CameraFollow script

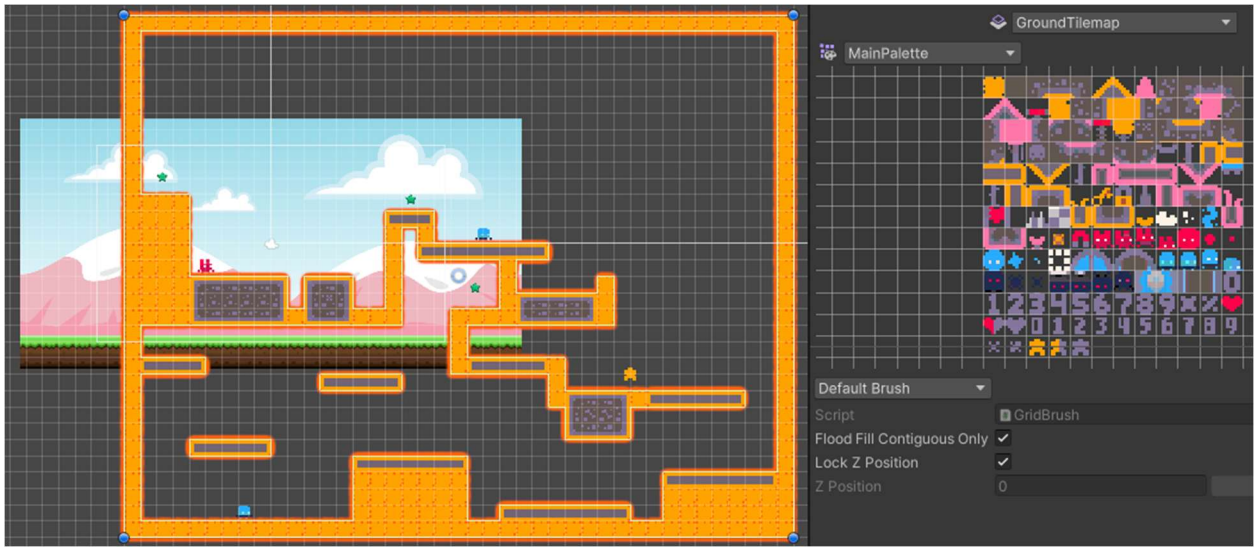


Demo Scene

Features:

- Fully functional example
- Built with Kenney's free assets
- Designed for clarity and customization

Edit the scene directly or use it as a foundation for your own levels.



Customization Tips

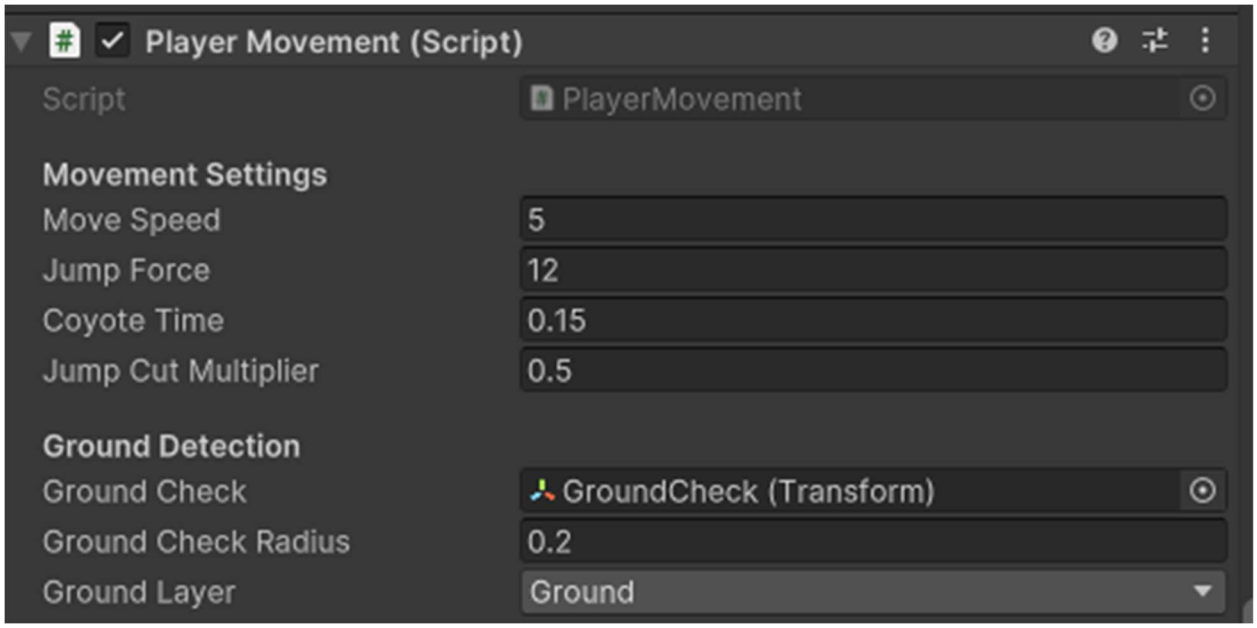
This framework is designed to be modular, allowing you to replace or extend any feature with minimal effort. Here are some practical ways to adapt the system for your own projects:

Player Controller

- **Movement Speed & Jump Force**
  Adjust these in the PlayerMovement component directly in the Inspector. Great for tweaking the "feel" of movement.
- **Animations**

Replace or add animations via the **Animator Controller**. You can plug in your own sprite sheet or even 3D characters if desired.

- **Extendability**
  Add mechanics like **double jump**, **dash**, or **wall slide** by expanding the PlayerMovement script. This code is organized for easy insertion of new states or abilities.



Enemies & Hazards

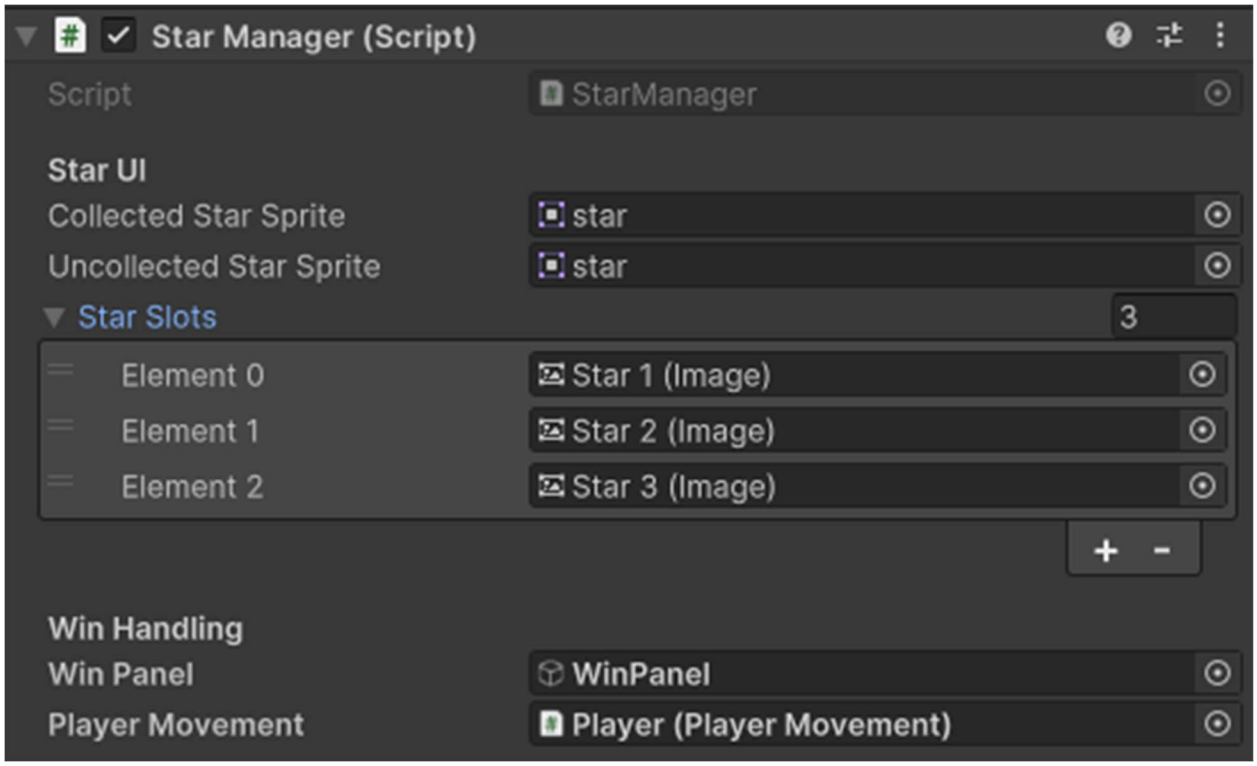
- **More Behavior Types**
  You can swap the EnemyPatrol with chase, idle, or ranged logic.
- **Damage Values**
  Modify TouchDamage.damageAmount per enemy to create stronger or weaker threats.
- **Custom Sprites**
  Just drop in a new sprite and adjust the scale. The AI logic remains unchanged.

Collectibles

- **Add New Star Locations**

Duplicate the star prefab and place it anywhere in the level.

- **Add More Stars (if needed)**
  - Increase the size of the starSlots array in StarManager
  - Update the win conditions logic if desired (e.g., require 5 stars instead of 3)

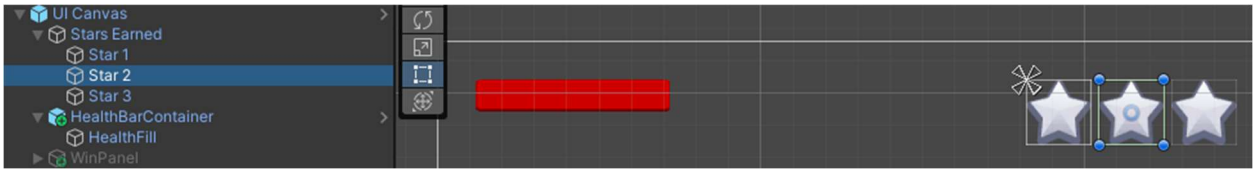


Checkpoints

- **Add Multiple Checkpoints**
  Duplicate the prefab and position them anywhere you want to allow progress-saving.
- **Customize Visuals**
  Swap sprite, add particle effects, or animations when activated.
- **Extend Logic**
  You can expand the checkpoint system to save more data (e.g., power-ups, score, inventory).

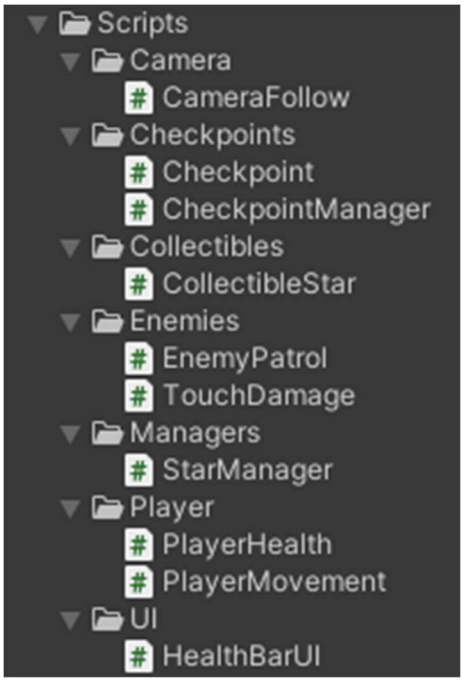
UI Customization

- **Replace UI Sprites**
  All UI elements (stars, panels, buttons) are swappable. Use your own icons, colors, or fonts.
- **Repositioning UI**
  Feel free to move or anchor the star counter, health bar, or win panel to new screen corners.



Scripting Overview

The codebase for **SolarSoft Platformer Free** is built to be clear, minimal, and modular. Each script handles one specific task, following the **single responsibility principle**. This makes it easy to replace, extend, or remove features as needed.



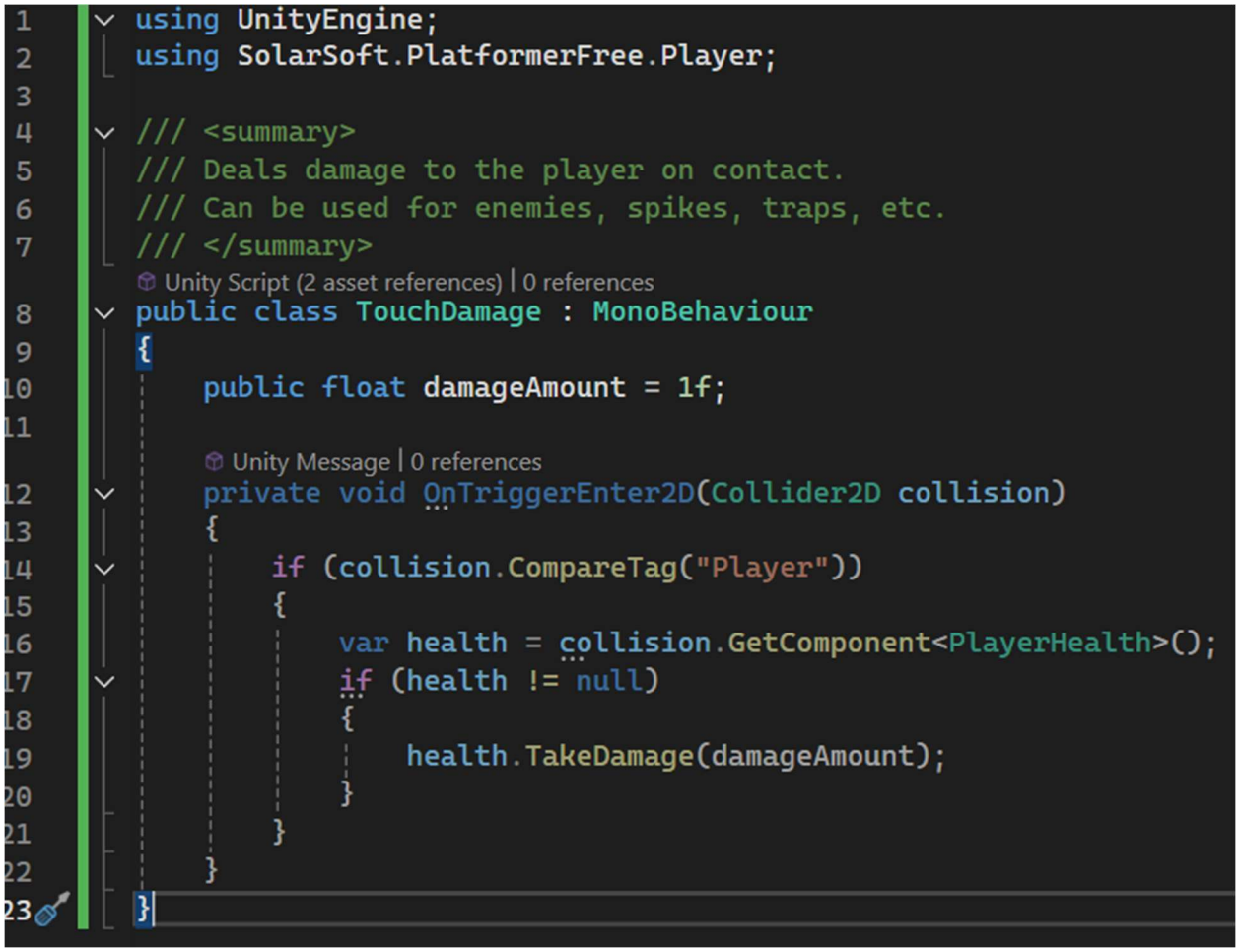
Script Breakdown

Script Name	Purpose
PlayerMovement.cs	Handles movement, jumping, coyote time, and flipping
PlayerHealth.cs	Manages health, damage, death animation, and respawning
HealthBarUI.cs	Updates the health bar fill visually based on player health
EnemyPatrol.cs	Moves an enemy between two points with optional flipping
TouchDamage.cs	Damages the player on contact (used for enemies or hazards)
Checkpoint.cs	Marks a location for respawn upon death
CheckpointManager.cs	Static class storing checkpoint state and position
StarManager.cs	Tracks collected stars, updates star UI, and handles win conditions
CollectibleStar.cs	Triggers StarManager when collected
CameraFollow.cs	Smoothly follows the player with optional offset

Architecture Highlights

- **Singleton Pattern:**
  StarManager uses a static Instance for global access. This keeps collectible logic simple and accessible from anywhere.
- **Modular Organization:**
  Each behavior (movement, health, AI, etc.) is in its own script. No script exceeds a single responsibility.
- **Serializable Fields:**
  Most values (speed, jump force, etc.) are exposed in the Inspector for non-programmers to tweak.

- **Safety Checks:**
  Null checks and simple conditionals prevent crashes if a component is missing or not assigned.



```
using UnityEngine;
using SolarSoft.PlatformerFree.Player;

/// <summary>
/// Deals damage to the player on contact.
/// Can be used for enemies, spikes, traps, etc.
/// </summary>
Unity Script (2 asset references) | 0 references
public class TouchDamage : MonoBehaviour
{
    public float damageAmount = 1f;

    Unity Message | 0 references
    private void OnTriggerEnter2D(Collider2D collision)
    {
        if (collision.CompareTag("Player"))
        {
            var health = collision.GetComponent<PlayerHealth>();
            if (health != null)
            {
                health.TakeDamage(damageAmount);
            }
        }
    }
}
```

Extending the Framework

SolarSoft Platformer Free is designed as a foundation. Its architecture allows for easy expansion without needing to modify the core systems drastically. Whether you're a beginner learning how systems connect or an indie developer looking to prototype a commercial product, this framework supports rapid iteration and modular development.

Common Extensions (Suggested Upgrades)

Here are some typical features you might consider adding to your own projects:

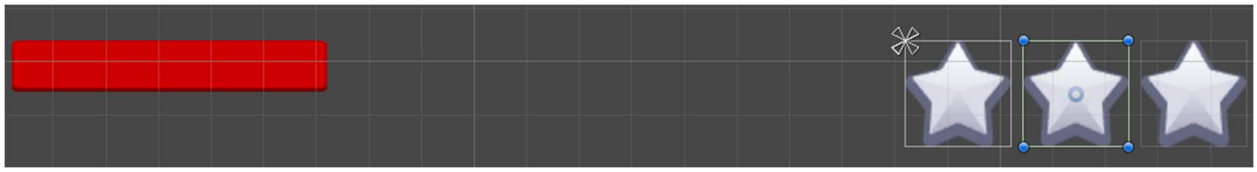
Feature	Example Extension Ideas
💥 **Combat System**	Add ranged or melee attacks, enemy health, and damage responses
🎯 **Enemy Variety**	Flying enemies, pathfinding, ranged attacks
🎁 **Power-Ups**	Temporary boosts (e.g., invincibility, speed, double jump)
🪙 **Currency & Shop**	Add coin collection, and a shop UI to spend coins
📋 **Level Selector**	Create a level hub with locked/unlocked levels and stars earned
🧠 AI Behavior Trees	Expand enemy intelligence with detection zones or patrol routines
🎮 Gamepad Input	Add full controller support via Unity's Input System
💾 Save System	Store collected stars, completed levels, and settings
🔊 Sound & Music	Add audio clips for actions, background music, and effects

Tips for Expansion

- **Use ScriptableObjects** to manage things like enemy stats, item types, or level data.
- **Create new scenes** for level design and use additive loading for menus or transitions.
- **Wrap new features** in clearly named namespaces to keep your code organized.
- **Follow naming conventions** and keep functions short and purposeful.
- **Keep core scripts untouched** where possible, extend them through composition or inheritance.

Asset & UI Overview

This section highlights the visual assets and UI elements used in **SolarSoft Platformer Free**, including how they are organized, where they're located, and how they tie into the project's systems.



Visual Assets

All visuals in this project use assets from Kenney.nl, except the background image, which is from OpenGameArt.org. These assets are free for personal, educational, and commercial use.

Main asset pack used:

- **Kenney Platformer Pack**
  Includes tilemaps, characters, UI elements, and Collectible items.

Asset Folder Path:

SolarSoft_Platformer_Free/Art/kenney_pico-8-platformer/

You may freely replace or expand on the existing art to customize the look of your game.

UI Elements Overview

Element	Description	File Locaation
Health Bar	Displays the player's current health	UI Canvas/HealthBarContainer → updates via HealthBarUI.cs
Star Counter	Shows how many stars have been collected	UI Canvas/Stars group → controlled via StarManager.cs
Win Panel	Shows when player collects all stars	UI Canvas/WinPanel → includes buttons for Restart/Exit

UI elements are created using Unity's built-in **Canvas** system. The background image is a **Sprite** attached as a child of the **Main Camera**, which ensures it scrolls naturally with the player's movement.

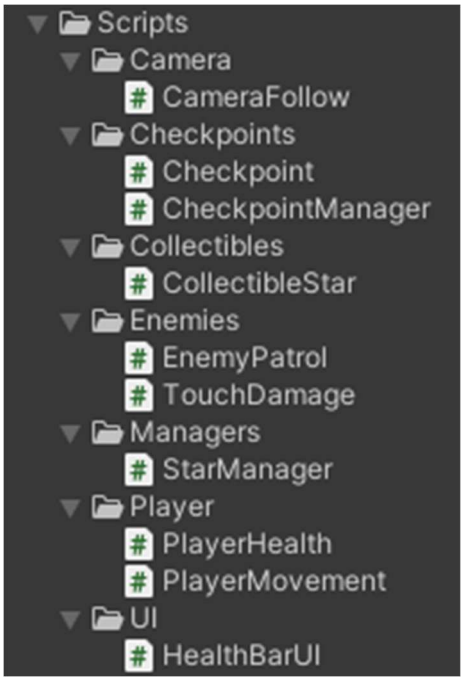
Customizing the UI

You can easily customize or restyle the UI:

- Change **panel backgrounds** and button styles in the Image components.
- Update **fonts**, colors, and transitions using Unity's **UI components**.
- Add sound effects, tooltips, or additional UI for pro features later.

Code Overview

This section introduces the core scripts that make up the **SolarSoft Platformer Free** framework. Each script is cleanly organized, well-commented, and designed for modularity and readability.



Folder Structure (Scripts)

- SolarSoft_Platformer_Free/
    - Scripts/
        - Camera/ → controls behavior such as smooth following.
        - Checkpoints/ → Manages player checkpoint activation and respawn logic.
        - Collectibles/ → Handles collectible stars and win conditions.
        - Enemies/ → Contains enemy patrol and damage logic.
        - Managers/ → Global game logic and singleton managers.
        - Player/ → Handles player movement, health, and animations.
        - UI/ → Simple UI logic such as health bar visuals.

Key Scripts

Script Name	Purpose
PlayerMovement.cs	Handles horizontal movement, jumping, and animations
PlayerHealth.cs	Tracks player health, triggers death and respawn logic
EnemyPatrol.cs	Moves enemies between two points, with flip animation
TouchDamage.cs	Damages the player on contact with enemies
StarManager.cs	Manages star collection, win condition, and win panel display
HealthBarUI.cs	Updates the health bar fill amount
CameraFollow.cs	Smoothly follows the player with an optional offset
Checkpoint.cs	Triggers checkpoint update when player enters area
CheckpointManager.cs	Static class that stores the most recent checkpoint position

Each script includes detailed summaries, serialized fields for easy editor access, and logic grouped by purpose. The code is safe to modify, expand, or repurpose in more advanced projects.

Tips for Developers

- Scripts are **decoupled** where possible to support replacement or upgrade.
- Code follows Unity best practices: Awake() for references, Start() for setup, and Update() / FixedUpdate() for logic.
- Use the **namespace structure** to organize new features when building Pro add-ons.

Gameplay Flow

SolarSoft Platformer Free offers a concise and engaging gameplay loop that showcases core platformer mechanics while remaining simple to understand and expand.

Start of Level

The player begins at a designated **start position**, with full health and no stars collected. The level is presented as a side-scrolling platform environment.

Player Movement & Exploration

Players control their character using:

- **Arrow Keys / A & D** to move left and right
- **Spacebar** to jump

Movement features include:

- **Smooth ground traversal**
- **Jumping with Coyote Time** (brief forgiveness after leaving ground)
- **Variable Jump Height** (shorter jump if button is released early)

Enemy Encounters

Enemies patrol between two points. If the player touches an enemy:

- The player takes damage
- If health reaches zero, a **death animation** plays
- Player is then:
    - **Respawned at the last checkpoint** (if available and activated)
    - **Returned to the start position** (if no checkpoint has been activated)

Checkpoint Activation

Checkpoints are triggered by walking into them. Once activated:

- The player will respawn at the checkpoint location after death
- Velocity is reset and full health is restored

Collecting Stars

There are **three stars** scattered throughout the level:

- When touched, each star disappears and updates the UI
- After all 3 stars are collected, a short delay occurs

Win Condition

Once all stars are collected:

- The **"You Win" panel** is displayed
- Player controls are disabled
- Two buttons appear:
    - **Restart:** Reloads the current scene
    - **Exit:** Quits the application (or Play Mode in the Editor)

Credits

SolarSoft Platformer Free was created by **SolarSoft Media, LLC.**, developed and documented by **Donald Faulknor**.

Asset Attributions

Kenney Assets

- Tilesets, sprites, and UI elements provided by Kenney.nl
- License: CC0 1.0 Universal (Public Domain)

Kenney assets are free to use in personal, educational, and commercial projects. Support the creator at https://kenney.nl if you find the assets useful.

OpenGameArt.org

- Background image provided by OpenGameArt.org
- License: CC0 1.0 Universal (Public Domain)

Final Notes

Thank you for using SolarSoft Platformer Free!

This framework was designed to empower developers of all skill levels to dive into 2D platformer creation with confidence. Whether you're a student learning Unity, a hobbyist experimenting with gameplay mechanics, or a studio building out a prototype, this package offers a clean foundation you can trust.

If you enjoy using this framework:

⭐ Consider leaving a review on the Unity Asset Store – it helps more than you think.

🔄 Watch for **SolarSoft Platformer Pro**, offering advanced features like multiple levels, enemies with behaviors, UI transitions, save systems, a special surprise, and more.

🛠️ Extend and modify the code – that's what it's here for. This package is made to grow with you.

Stay creative, and keep building amazing things.

— **Donald Faulknor**

Founder, SolarSoft Media, LLC

📬 support@solarsoftmedia.com