🎮 Beginner Game Toolkit – README

Welcome to the **Beginner Game Toolkit**!
This toolkit gives you **ready-made mechanics** to build endless runners, tap games, or simple prototypes **without needing to write code**.
All the scripts are **modular** and use the **Inspector** and **events** to keep things beginner-friendly.

🪄The Magic Ingredient: Events (No Code Needed)

Example:“Player Crosses the Finish Line → Celebration Chain!”

> When your player hits the goal, the game celebrates — confetti, victory sound, screen popup, and even a color flash.

🏁 Step 1. The Starting Point — The Goal Object

The **GoalTrigger** (a simple collider with a trigger) is the event sender.
When the player touches it, it tells the GameStateManager to fire an event called `OnGameWin`.

So it’s like the goal saying:

> “Someone crossed the finish line! Time to celebrate!”

🎆 Step 2. The Listeners (All the Fun Stuff!)

You can connect **as many listeners as you want** to that one event.
Let’s make it exciting:

Listener	What It Does	Component

🎊 ConfettiSpawner	Spawns confetti particle effect	`EffectTrigger.PlayEffect()`
🔊 SoundManager	Plays victory music	`SoundManager.PlayWinSound()`
🖥️ WinScreen	Shows the “You Win!” screen	`WinScreen.Show()`
🌈 CameraFlash	Flashes screen color	`EffectTrigger.PlayEffect()`

💥 Done.
When the player collides with the goal, all those effects happen **instantly**, all triggered by a single event.

🚀 Getting Started

1. Create a new Unity project.
2. Import the **Toolkit unitypackage**
3. Drop into and hook together the prefabs you need in your scene.
4. Hit Play — you should have a running prototype!

🧩 Core Systems

1. Input Manager

- Detects inputs from Space, A, D

✅ How to use:

- Place `InputManager.prefab` in the scene.

---

2. Player Movement

Swerve Controller

- Two modes:
    - **Pan**: Smooth left/right panning.
    - **Turn**: 90° snapping turns (Temple Run style).

✅ How to use:

- Add `SwerveController.cs` to the Player (if not already there).
- In the Inspector → choose **Swerve Mode**.

Endless Runner Forward Movement

- Keeps the player moving forward automatically.
- Can be paused/resumed with events (`StartRunning()` / `StopRunning()`).

---

3. Collision & Gameplay

- **Obstacles:** Player detects collision → triggers `GameStateManager.LoseGame()`.
- **Pickups:** Each pickup prefab has its own script → adds score when collected.
- **Goal Line:** Goal prefab detects collision with Player → triggers `GameStateManager.WinGame()`.

---

4. Score System

- `ScoreManager.cs` keeps track of score and highscore.

✅ How to use:

- Place `ScoreManager.prefab` in the scene.
- Hook up `ScoreText in ScoreUI` → Text field on ScoreManager called `ScoreText`.

---

5. Game State Manager

- Controls Win/Lose conditions.
- Fires `OnGameWin` and `OnGameLose` events.
- Also fires `OnGameRespawnAtCheckpoint, OnGameRestart, OnGameNextLevel`

✅ How to use:

- Place `GameStateManager prefab` in the scene.
- Hook up UI screens to the events (f.x. WinScreen.Show, LoseScreen.Show).
- Hook up player movement stopping or starting, restarting, next or perhaps main menu scene (SceneTransitionManager.ReloadCurrentScene/LoadNextScene)

---

6. UI Screens

- **WinScreen.prefab** → shows when `OnGameWin` is fired if you've hooked it up there.
- **LoseScreen.prefab** → shows when `OnGameLose` is fired if you've hooked it up there.

✅ Tip:
Keep them **active in the scene** but with their **Canvas disabled**.
That way, they'll still listen to events and can activate themselves.

🛠️ Extra Features

Scene Transition Manager

- Handles moving between scenes.
- Can also restart or quit the game.

✅ Example button hookups:

- `LoadNextScene("Level1")` → start the game from Main Menu.
- `ReloadCurrentScene()` → restart from LoseScreen.

Main Menu

Prefab: `MainMenu.prefab`

- Play button → calls `SceneTransitionManager.LoadScene("Level_1")`.
- Quit button → calls `SceneTransitionManager.QuitGame()`.

✅ Example button hookups:

- `LoadScene("Level_2")` → start Level 2 from Main Menu using a Level 2 button.
- `LoadScene("Level_3")` → start Level 3 from Main Menu using a Level 3 button..

Sound Manager

- Plays background music and sound effects.

- Persistent across scenes. Meaning it can keep playing the music across scenes and won't restart the music even if everything else is rebuilt in that scene.
- You can also make small `SoundEffectPlayer` prefabs for one-shot sounds and connect them to your `EffectTrigger` events.

✅ How to use:

- Drop `SoundManager.prefab` into the first scene.
- Assign a music clip to `musicSource`.
- Call `SoundManager.Instance.PlaySFX()` for button clicks, collisions, etc. F.x. through events.

(Optional) Procedural Level Generator

- Spawns random tiles as the player runs forward. Only spawns tiles in a forward direction. So it only works with forward movement player controls.
- Old tiles are deleted to save memory.

✅ How to use:

- Add `ProceduralLevelGenerator.cs` to an empty GameObject.
- Assign the Player and a list of Tile Prefabs.
- Hit Play → endless level generation!

🔀 Collision Responsibility (Who Detects What?)

- **Player** → detects bad collisions (obstacles → death).
- **Pickup prefab** → detects good collisions (coins → reward).

- **Goal prefab** → detects win state (finish line → win).

This way:

- Dragging in a new obstacle prefab → instantly works with the Player's collision.
- Dropping in a new pickup prefab → instantly adds score/powerup without editing the Player script.

🎨 Recommended Workflow for Beginners

1. Start with `MainMenu.prefab` + `SceneTransitionManager.prefab`.
2. Add a `GameScene`:
    - Drop in `InputManager.prefab`, `GameStateManager.prefab`, `ScoreManager.prefab`.
    - Add your Player prefab with `SwerveController` + Runner script.
    - Add pickups, obstacles, and (optional) GoalLine prefab.
    - Add and hook up `WinScreen.prefab` and `LoseScreen.prefab`.
    - Add `SoundManager.prefab`.
3. Press Play → you should have a working endless runner with scoring, win/lose, and menus!

🏆 Tips for Experimenting

- Change **Swerve Mode** between `Pan` and `Turn`.
- Create new pickups (duplicate the coin prefab and change the reward).

- Build 3 levels in 3 different scenes. **Or** try out the `ProceduralLevelGenerator` using different tiles for different environments.

- Add sound FX to UI buttons using the `SoundManager`.

- There are more things in the toolkit. Experiment and have fun with them <3

That's it! 🎉 With this toolkit you can mix & match features to create endless runners, tap-to-win games, or prototype new mechanics without touching much code.