
Unity 2D
Basics
Part 4: Events and
Simple Movement

Events and Simple Movement

- In the previous part, we introduced a sprite as an example of a 2D GameObject. It has Transform (where-to-draw) and a Sprite Renderer (what-to-draw) components.
- We want to get to the goal of "press a button and a thing on the screen moves."

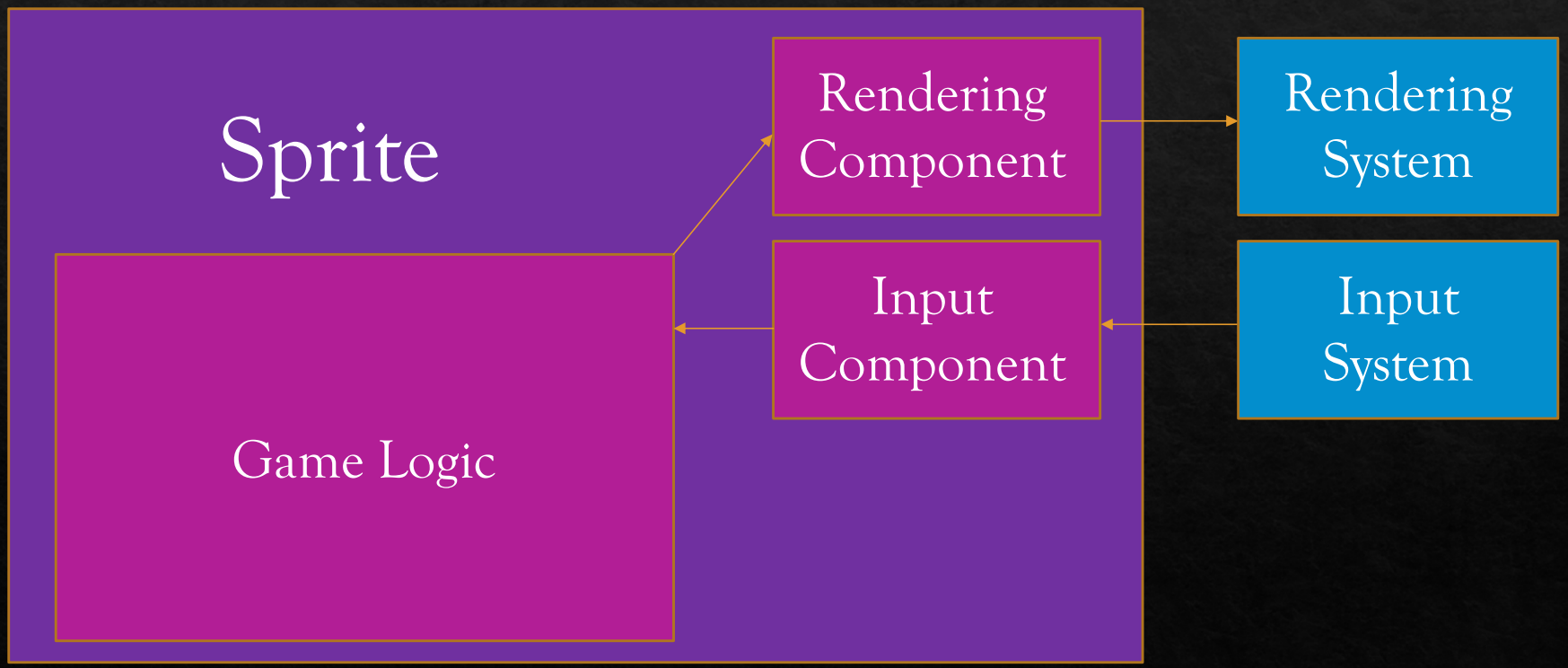
Events and Simple Movement (cont.)

Components	
✔	Rendering
✔	Input
✔	Game Logic (Behavior Script)

Goal: *An input system sends events to game logic to manipulate the position of the GameObject.*

We finally have the three components we need: rendering, input, and game logic.

Sprite


Sprite
Game Logic
Rendering Component
Input Component
Rendering System
Input System

Events and Simple Movement (cont.)

- When we write code in Unity, we use the programming language C#. This is written in Visual Studio.
- We write behavior scripts that react to events as different systems run in Unity.

Events and Simple Movement (cont.)

- Common C# Terms:
  - Class (blueprints for future objects)
    - Properties (values in the class)
    - Methods (how objects relate to each other)
- Changes in Unity: *behavior scripts receive events and react through their methods.*

Events and Simple Movement (cont.)

- Default methods and their ordering (order of execution):
  - Start() – part of Initialization (before systems start)
  - Update() – part of Game Logic
- We can always add more methods to listen for other events!

Events and Simple Movement (cont.)

- Debugging is an important part of development. One of the most common tools in Unity is the Debug.Log() method. We can use it to see messages in the Console Window in the Unity Editor.

Events and Simple Movement (cont.)

- As components themselves, all behavior scripts have access to *gameObject*, a property representing the GameObject the component is a part of.
- All GameObjects have a Transform component. This also means that each has a property called *gameObject.transform*. This gives access to changing values via methods.

Events and Simple Movement (cont.)

- To listen for InputSystem events, we also must add `using UnityEngine.InputSystem;` to the code. This will add the C# events to our project and allow us to use them in our code.
- Once added, we can then access any of our defined events.