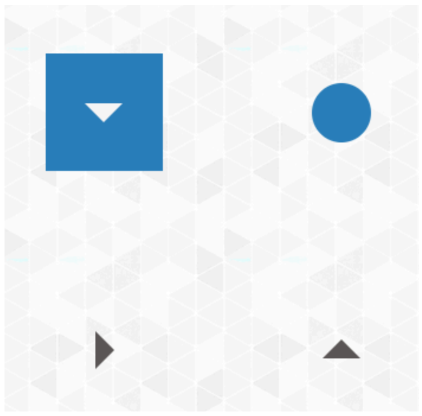
Game System Breakdown - Game about Squares

Rotate Mechanic

This rotate mechanic solves an issue for both the designers and the players. Without the rotate mechanic, it would be both impossible to beat most levels or it would limit the game designer to only a few paths that get repetitive and become too easy for the player. The rotate mechanic introduces a new variety of challenges since they could be placed anywhere and force the player to think about which rotation each square might need and in what order.



The Ingredients

For the rotate mechanic in particular, the objects are the square and the rotate key as shown above. The ingredients required to put these together is at least one script for the square's movement, and one (which can be combined into the square's movement script) that'll detect collision with the rotate key. A controller or parent class could also be used to control the base movement of the squares, although all of the squares behave the same in this game.

Pseudocode

Prefab - Squares & rotate keys (since we need multiple of both for each level)
Rigidbody2D & Colliders for both Square & Rotate Key (Dynamic for Square, Static for Rotate Key)
Small function in update for collision (could be under the square or the rotate key)
A function in rotate key that will adjust the rotation of the square upon collision or trigger (since only the square will interact with the rotate key).
A static variable for square's movement and one for rotation

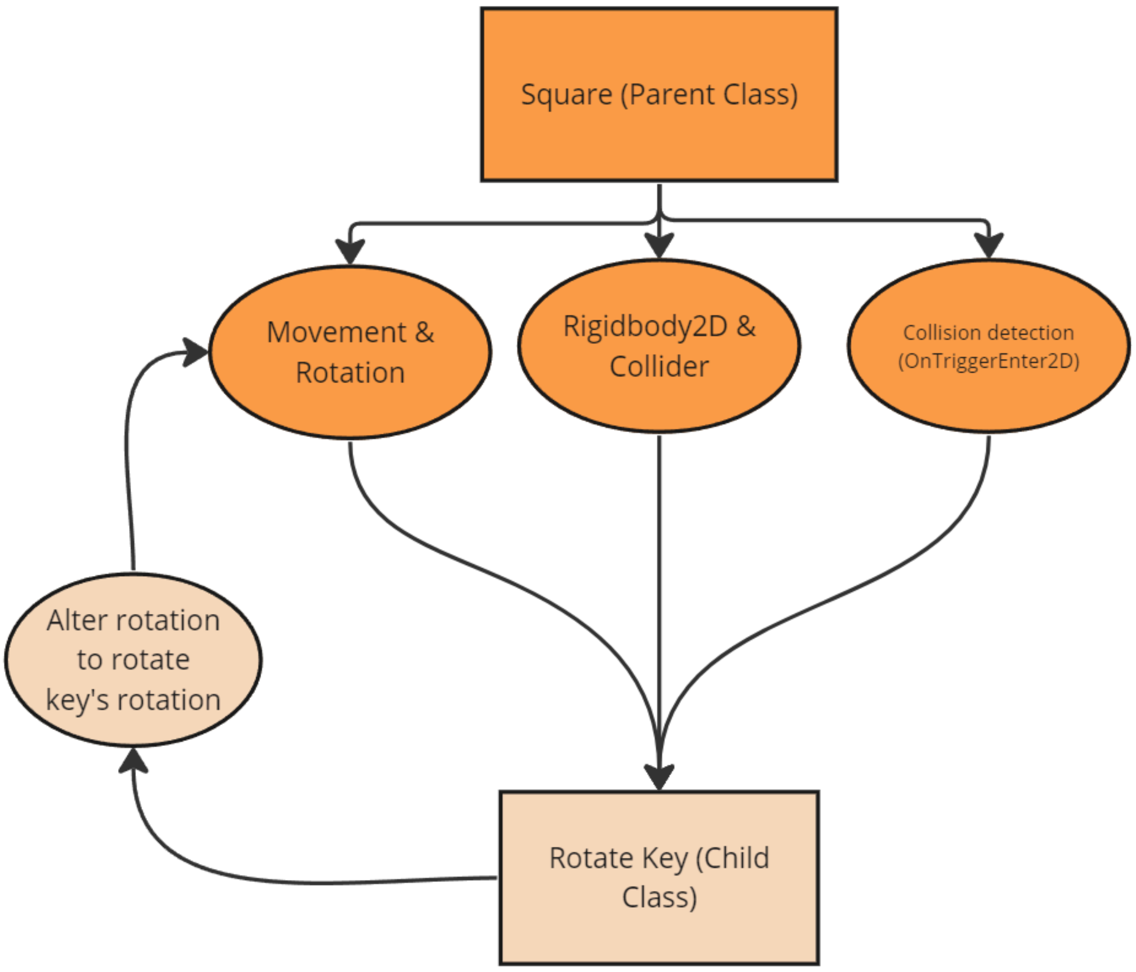
Composition

Given that the square would be the parent, there are only two main functions that would relate to the rotate key: one for movement, and potentially one for rotation as well. The movement function would ensure that the square moves a single space while rotation determines which direction the square is facing. Collisions would be detected via OnTriggerEnter2D which can interact with the "end point" and the rotate key.

Inheritance

It's hard to say whether or not inheritance could be used to replicate this in the typical sense. The rotate mechanic could work just fine by the two objects communicating with one another

via SendMessage or on trigger. More so, the rotate key and square don't resemble one another in what might be their script.
The rotate key would be a child of the square (parent class) and wouldn't use its movement function but it could use its rigidbody2d, collider and communicate with the script in control of the square's rotation. The movement function in the parent class would just remain privated. The only part that the rotate key needs to inherit is in regards to the rotation, and it only needs to be rotated by 90 upon trigger.


Square (Parent Class)
Movement & Rotation
Rigidbody2D & Collider
Collision detection (OnTriggerEnter2D)
Alter rotation to rotate key's rotation
Rotate Key (Child Class)