

# **What I did In**

---

# **my 5+ years**

# **of experience**



Alaa Abusarah  
@AlaaAbusarah

# Behavior Cup

AI-Framework for Unity Engine! ★  
based on Behavior Trees patterns

The screenshot displays the Behavior Cup AI Framework interface. On the left, the **Inspector** window shows settings for a script named "FOV". The "Settings" section includes fields for Radius (10), Angle (360), and Offset (X 0, Y 1, Z 0). It also lists Target Mask (Character) and Obstruction Mask (Ground). Under the "Tag" section, "Use Tag" is checked with "Tag" set to "Player". The "Outputs" section shows "Saving Key" as "Target". On the right, the **Minimap 1.00x** window shows a map with several gray blocks and a green player icon. In the center, a detailed Behavior Tree diagram is shown:

```
graph TD; Root[Root Node] --> Repeat[Repeat]; Repeat --> Sequencer[Sequencer]; Sequencer --> FOV[FO V]; Sequencer --> Rotate[Rotate]; Sequencer --> Timer[Timer]; Timer --> Instantiate[Instantiate On Transform]
```

The tree starts with a **Root Node**, which branches into a **Repeat** node. The **Repeat** node branches into a **Sequencer** node. The **Sequencer** node branches into three parallel nodes: **FO V**, **Rotate**, and **Timer**. The **Timer** node has a single outgoing connection labeled **Instantiate On Transform**.

# **Highway Hajwala**

**Mobile drifting game published on  
google play**



THIS IS MY FAVORITE 💪

# Field Of View

Tool for detecting objects within a specified radius and angle

```
0 references
public List<T> Field<T>(string tag = null)
{
    List<T> value = new List<T>();

    //Find all objects in range.
    Collider[] rangeChecks = Physics.OverlapSphere(transform.position + offset, radius, targetMask);

    for (int i = 0; i < rangeChecks.Length; i++)
    {
        Transform target = rangeChecks[i].transform;
        Vector3 directionToTarget = (target.position - transform.position).normalized;

        if (tag != null && target.tag != tag) continue;//Check for condition tag.

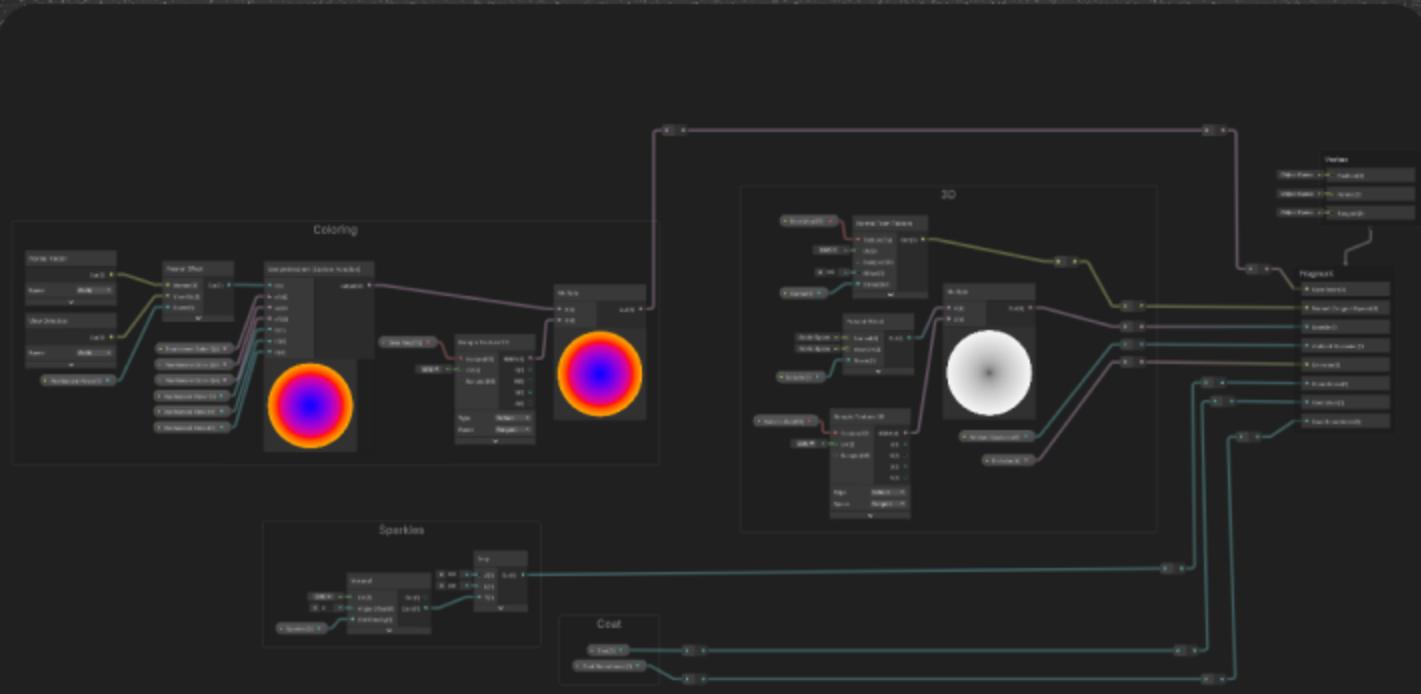
        if (Vector3.Angle(transform.forward, directionToTarget) < angle / 2)
        {
            //Object in view angle.

            float distanceToTarget = Vector3.Distance(transform.position, target.position);
            if (!Physics.Raycast(transform.position, directionToTarget, distanceToTarget, obstructionMask))
            {
                //No obstruction in direction.

                T t;
                target.TryGetComponent<T>(out t);
            }
        }
    }
}
```

# Painting Shader

Realistic car painting shader for  
Unity mobile games



# Car Controller

Tool for helping developers for  
prototyping car controller



# **2D Platforming**

---

**Old game but still gold, Before 3  
years**

**TAP TO START**



**AlaaAbusarah.com**  
**Work@AlaaAbusarah.com**  
**+962781145276**