Behavior Designer - Formations Pack

Table of Contents

Behavior Designer - Formations Pack

The Behavior Designer – Formations Pack includes 14 tasks focused on group formations. The default set of Formations Pack tasks use Unity's navigation mesh to traverse the world. The Formations Pack doesn't do the actual movement – it instead sets the destination for the underlying pathfinding implementation (Unity's NavMesh, A* Pathfinding Project, etc).

Integrations

You can download the Formations Pack integrations from this page. After you have imported an integration you can start to use the integration tasks by adding them to your behavior tree. For example, if you want to use the A* Pathfinding Project version of the grid task you would add the task located under Actions -> Formations -> A* Pathfinding Project -> Grid.

New Formation Task

As we hear more suggestions from the community the number of formation tasks will continue to grow. However, you may want to add your own custom formation and this can easily be accomplished with the Fomration Pack API. The first step in developing your own formation task is to inherit your task from the GroupFormation base class. This will take care of most of the work for you and all you have to do is specify where you want each of your units to be positioned. The following methods can be overridden:

```
/// Virtual method to allow the formation tasks to specify a target
position.
/// index: The index of the group member.
/// zLookAhead: The z distance to look ahead of the target position.
/// Returns: The position to move to, in world space.
protected Vector3 TargetPosition(int index, float zLookAhead)

/// Adds the agent to the formation group.
/// agent: The agent to add.
/// Returns: The index of the agent within the group.
protected void AddAgentToGroup(Behavior agent, int index)

/// Removes the agent from the group.
/// agent: The agent to remove.
/// Returns: The index of the agent removed from the group.
protected int RemoveAgentFromGroup(Behavior agent)
```

TargetPosition is the method that will position your agent. The index parameter specifies the index of the agent within the formation. zLookAhead specifies the distance to look ahead of the target position. As a basic example, the following method will form a line where each agent is two units apart on the relative x axis:

```
protected override Vector3 TargetPosition(int index, float zLookAhead)
{
   return transforms[0].TransformPoint(index * 2, 0, zLookAhead);
}
```

The transforms array is a protected array belonging to the parent class. It contains a list of all of the group transforms. In this example we are getting the relative offset from the first transform, which is always the leader transform. As the index increases the distance between the agent and the leader will also increase on the x axis which will form a line. zLookAhead is used on the relative z axis to prevent the agent from always being within stopping distance from its target.

AddAgentToGroup and RemoveAgentFromGroup are called when the agent is added or removed from the group. This allows for initialization/destruction if the formation task requires knowledge ahead of time of how many agents are in the group.