UnitFormation

Table of Contents

Script Reference

Unit Formation

Demo

Scripts

FormationUnit.cs

Classes

FormationUnit

```
public class FormationUnit
```

Simple unit for demonstration of movement to target position with built-in Unity AI system. It also faces the angle of the formation after it reaches its destination.

Variables

agent

```
private NavMeshAgent agent
```

facingAngle

```
private float facingAngle
```

faceOnDestination

```
private bool faceOnDestination
```

rotationSpeed

```
[SerializeField,Tooltip("Speed with which the unit will rotate
towards the formation facing angle.")]
private float rotationSpeed
```

FacingRotationEnabled

```
[HideInInspector]
public bool FacingRotationEnabled
```

Specifies if rotating towards the facing angle is enabled. Set this to 'false' if you wish to manually handle synced rotation of units in rotation.

IsWithinStoppingDistance

```
public bool IsWithinStoppingDistance
```

Methods

Start

```
private void Start()
```

Update

```
private void Update()
```

SetTargetDestination

```
public void SetTargetDestination(
     Vector3 newTargetDestination,
     float newFacingAngle)
```

SyncedFormationRotation.cs

Classes

SyncedFormationRotation

```
public class SyncedFormationRotation
```

Demonstration component for syncing direction change of all units. This will prevent units from facing formation angle until all units have reached their destination.

Variables

formationControls

```
private UnitFormationControls formationControls
```

formationUnits

```
private List<FormationUnit> formationUnits
```

Methods

Start

```
void Start()
```

Update

```
void Update()
```

UnitFormationControls.cs

Classes

UnitFormationControls

```
public class UnitFormationControls
```

Variables

units

```
public List<GameObject> units
```

List of units in the scene

groundLayerMask

```
[SerializeField]
private LayerMask groundLayerMask
```

Specifies the layer mask used for mouse point raycasts in order to find the drag positions in world/scene.

LineRenderer

```
[SerializeField]
private LineRenderer LineRenderer
```

Specifies the line renderer used for rendering the mouse drag line that indicates the unit facing direction.

UnitCountSlider

```
[SerializeField]
private Slider UnitCountSlider
```

Specifies the unit count that will be generated for the scene. May be adjusted in realtime.

UnitSpacingSlider

```
[SerializeField]
private Slider UnitSpacingSlider
```

Specifies the unit spacing that will be used to generate formation positions.

RectangleColumnCountSlider

```
[SerializeField]
private Slider RectangleColumnCountSlider
```

UnitCountText

```
[SerializeField]
private Text UnitCountText
```

Specifies the Text used to represent the unit count selected by UnitCountSlider.

UnitSpacingText

```
[SerializeField]
private Text UnitSpacingText
```

Specifies the Text used to represent the unit spacing selected by UnitSpacingSlider.

RectangleColumnCountText

```
[SerializeField]
private Text RectangleColumnCountText
```

UnitPrefab

```
[SerializeField]
private GameObject UnitPrefab
```

currentFormation

```
private IFormation currentFormation
```

isDragging

```
private bool isDragging
```

pivotInMiddle

```
private bool pivotInMiddle
```

noiseEnabled

```
private bool noiseEnabled
```

unitCount

```
private int unitCount
```

rectangleColumnCount

```
private int rectangleColumnCount
```

unitSpacing

```
private float unitSpacing
```

Methods

Start

```
private void Start()
```

Update

```
private void Update()
```

HandleMouseDrag

```
private void HandleMouseDrag()
```

ApplyCurrentUnitFormation

```
private void ApplyCurrentUnitFormation()
```

SetUnitFormation

```
private void SetUnitFormation(
     IFormation formation)
```

OnNoiseToggleChanged

```
public void OnNoiseToggleChanged(
     bool newState)
```

OnPivotToggleChanged

```
public void OnPivotToggleChanged(
     bool newState)
```

LineFormationSelected

```
public void LineFormationSelected()
```

CircleFormationSelected

```
public void CircleFormationSelected()
```

TriangleFormationSelected

```
public void TriangleFormationSelected()
```

ConeFormationSelected

```
public void ConeFormationSelected()
```

RectangleFormationSelected

```
public void RectangleFormationSelected()
```

UpdateRectangleColumnCountText

```
public void UpdateRectangleColumnCountText()
```

UpdateUnitCountText

```
public void UpdateUnitCountText()
```

UpdateUnitSpacing

```
public void UpdateUnitSpacing()
```

ReinstantiateFormation

```
private void ReinstantiateFormation()
```

Instantiates a new formation based on the current type with the new configurations applied from UI.

Scripts

FormationPositioner.cs

Namespaces

TRavljen.UnitFormation

```
namespace TRavljen.UnitFormation
```

Classes

FormationPositioner

```
public class FormationPositioner
```

Class responsible for providing unit positions in formation on a target position facing the respective angle. Use method 'GetAlignedPositions' when you are manually calculating facing angle of the formation. Use method 'GetPositions' when you wish for angle calculation to be done manually. It will be calculated based on magnitude of the position change and the angle from units center position to the new target position.

Methods

GetAlignedPositions

```
public static List<Vector3> GetAlignedPositions(
     int unitCount,
     IFormation formation,
     Vector3 targetPosition,
     float targetAngle)
```

Returns aligned units formation positions that are facing the passed angle.

unitCount: Amount of units in formation.
formation: Formation that units will position in.
targetPosition: Position of the formation.
targetAngle: Facing angle for the formation.

Returns: Returns aligned positions of the units in formation.

GetPositions

```
public static UnitsFormationPositions GetPositions(
     List<Vector3> currentPositions,
     IFormation formation,
     Vector3 targetPosition,
     float rotationThreshold = 4.0f)
```

Finds new positions for the passed positions and the formation. If distance from current positions center is less than rotation threshold, units formation will not be rotated around the target. New rotation angle is calculated from center position of all current positions and the target positions.

currentPositions: Current unit positions.
formation: Formation used on units
targetPosition: Position to where the units will be moved.
rotationThreshold: Threshold used to specify when the unit formation should be rotated around target position (pivot).

Returns: Returns list of the new unit positions and their new facing angle

RotatePointAroundPivot

```
private static Vector3 RotatePointAroundPivot(
     Vector3 point,
     Vector3 pivot,
     Vector3 angles)
```

UnitsFormationPositions

```
public struct UnitsFormationPositions
```

Data structure that represents the units new formation positions and angles.

Constructors

UnitsFormationPositions

```
public  UnitsFormationPositions(
     List<Vector3> unitPositions,
     float finalRotation)
```

Variables

UnitPositions

```
public List<Vector3> UnitPositions
```

Specifies the new positions that units can move to new formation.

FacingAngle

```
public float FacingAngle
```

Specifies the units facing angle (loot at direction) for the new position.

UnitFormationHelper.cs

Namespaces

TRavljen.UnitFormation

```
namespace TRavljen.UnitFormation
```

Classes

UnitFormationHelper

```
public static class UnitFormationHelper
```

Methods

ApplyFormationCentering

```
public static void ApplyFormationCentering(
     ref List<Vector3> positions,
     float rowCount,
     float rowSpacing)
```

Applies offset to the Z axes on positions in order to move positions from pivot in front of formation, to pivot in center of the formation.

positions: Current positions, method will update the reference values.
rowCount: Row count produced with formation.
rowSpacing: Spacing between each row.

GetNoise

```
public static Vector3 GetNoise(
     float factor)
```

Generates random "noise" for the position. In reality takes random range in the offset, does not use actual Math noise methods.

factor: Factor for which the position can be offset.

Returns: Returns local offset for axes X and Z.

Formations

CircleFormation.cs

Namespaces

TRavljen.UnitFormation.Formations

```
namespace TRavljen.UnitFormation.Formations
```

Classes

CircleFormation

```
struct CircleFormation
```

Formation that positions units in a circle with specified angle and spacing between units.

Constructors

CircleFormation

```
public  CircleFormation(
     float spacing,
     float circleAngle = 360f)
```

Instantiates circle formation.

spacing: Specifies spacing between units in cricle
circleAngle: Specifies angle for units to be placed, 360 degree means that the units will go entire path around the circle and 180 degree angle means that only half of the circle will be formed.

Variables

spacing

```
private float spacing
```

circleAngle

```
private float circleAngle
```

Methods

GetPositions

```
public List<Vector3> GetPositions(
     int unitCount)
```

ConeFormation.cs

Namespaces

TRavljen.UnitFormation.Formations

```
namespace TRavljen.UnitFormation.Formations
```

Classes

ConeFormation

```
public struct ConeFormation
```

Formation that positions units in a cone shape with specified spacing.

Constructors

ConeFormation

```
public  ConeFormation(
     float spacing,
     bool pivotInMiddle = true)
```

Instantiates cone formation.

spacing: Specifies spacing between units.
pivotInMiddle: Specifies if the pivot of the formation is in the middle of units. By default it is in first row of the formation. If this is set to true, rotation of formation will be in the center.

Variables

spacing

```
private float spacing
```

pivotInMiddle

```
private bool pivotInMiddle
```

Methods

GetPositions

```
public List<Vector3> GetPositions(
     int unitCount)
```

IFormation.cs

Namespaces

TRavljen.UnitFormation.Formations

```
namespace TRavljen.UnitFormation.Formations
```

Classes

IFormation

```
public interface IFormation
```

Defines the contact that all formations must implement. Formation should be generated or provided on the fly by calling GetPositions(int).

Methods

GetPositions

```
List<Vector3> GetPositions(
     int unitCount)
```

LineFormation.cs

Namespaces

TRavljen.UnitFormation.Formations

```
namespace TRavljen.UnitFormation.Formations
```

Classes

LineFormation

```
public struct LineFormation
```

Formation that positions units in a straight line with specified spacing.

Constructors

LineFormation

```
public  LineFormation(
     float spacing)
```

Instantiates line formation.

spacing: Specifies spacing between units.

Variables

spacing

```
private float spacing
```

Methods

GetPositions

```
public List<Vector3> GetPositions(
     int unitCount)
```

RectangleFormation.cs

Namespaces

TRavljen.UnitFormation.Formations

```
namespace TRavljen.UnitFormation.Formations
```

Classes

RectangleFormation

```
public struct RectangleFormation
```

Formation that positions units in a rectangle with specified spacing and maximal column count.

Constructors

RectangleFormation

```
public  RectangleFormation(
     int columnCount,
     float spacing,
     bool centerUnits = true,
     bool pivotInMiddle = false)
```

Instantiates rectangle formation.

columnCount: Maximal number of columns per row (there are less rows if number of units is smaller than this number).
spacing: Specifies spacing between units.
centerUnits: Specifies if units should be centered if they do not fill the full space of the row.
pivotInMiddle: Specifies if the pivot of the formation is in the middle of units. By default it is in first row of the formation. If this is set to true, rotation of formation will be in the center.

Variables

spacing

```
private float spacing
```

centerUnits

```
private bool centerUnits
```

pivotInMiddle

```
private bool pivotInMiddle
```

Properties

ColumnCount

```
public int ColumnCount
     { get; }
```

Returns the column count which represents the max unit number in a single row.

Methods

GetPositions

```
public List<Vector3> GetPositions(
     int unitCount)
```

TriangleFormation.cs

Namespaces

TRavljen.UnitFormation.Formations

```
namespace TRavljen.UnitFormation.Formations
```

Classes

TriangleFormation

```
public struct TriangleFormation
```

Formation that positions units in a triangle with specified spacing.

Constructors

TriangleFormation

```
public  TriangleFormation(
     float spacing,
     bool centerUnits = true,
     bool pivotInMiddle = false)
```

Instantiates triangle formation.

spacing: Specifies spacing between units.
centerUnits: Specifies if units should be centered if they do not fill the full space of the row.
pivotInMiddle: Specifies if the pivot of the formation is in the middle of units. By default it is in first row of the formation. If this is set to true, rotation of formation will be in the center.

Variables

spacing

```
private float spacing
```

centerUnits

```
private bool centerUnits
```

pivotInMiddle

```
private bool pivotInMiddle
```

Methods

GetPositions

```
public List<Vector3> GetPositions(
     int unitCount)
```