NameSpace DaiMangou.ProRadarBuilder.Editor

RadarBlips2D Class Reference

Public Attributes

- bool **Instanced**
  checks if all blips have ben instanced
- bool **IsActive**
  check if the blip is set turned on or off
- bool **ShowBLipSettings**
  INTERNAL USE ONLY
- bool **ShowSpriteBlipSettings**
  INTERNAL USE ONLY
- bool **ShowPrefabBlipSettings**
  INTERNAL USE ONLY
- bool **IsTrackRotation**
  Determines if the blip will be tracking the rotation of its target
- bool **BlipCanScleBasedOnDistance**
  Determines if th blips can scale by distance
- bool **ShowGeneralSettings**
  INTERNAL USE ONLY
- bool **ShowAdditionalOptions**
  INTERNAL USE ONLY
- bool **AlwaysShowBlipsInRadarSpace**
  determines if the blip should always remeing inside the radar
- Sprite **icon** = new Sprite()
  The blip icon if the blip is a sprite
- string **State** = ""
  INTERNAL USE ONLY
- string **Tag** = "Untagged"
  INTERNAL USE ONLY
- Material **SpriteMaterial**
  The material used for the sprite blip
- Color **colour** = new Color(1F, 0.6F, 0F, 0.8F)
  The colour of the sprite blip
- float **BlipSize** = 1
  The size of the blip
- float **DynamicBlipSize** = 0.05f
  The constant minimul scale which the blips are scaled to
- float **BlipMinSize** = 1
  The minimum size of the blip
- float **BlipMaxSize** = 2

The maximum size of the blip

- int **Layer** = 0
  INTERNAL USE ONLY
- Transform **prefab**
  Prefab blip
- List< GameObject > **gos** = new List<GameObject>()
  A list of the objects being tracked
- List< Transform > **RadarObjectToTrack** = new List<Transform>()
  A list of the actual blips you see in your radar
- **CreateBlipAs _CreateBlipAs**
  Determines what the blip should be created as , prefab or sprite

RadarCenterObject2D Class Reference

Public Attributes

- bool **Instanced**
  checks if all blips have ben instanced
- bool **IsActive**
  check if the blip is set turned on or off
- bool **ShowCenterBLipSettings**
  INTERNAL USE ONLY
- bool **ShowSpriteBlipSettings**
  INTERNAL USE ONLY
- bool **ShowPrefabBlipSettings**
  INTERNAL USE ONLY
- bool **IsTrackRotation**
  Determines if the blip will be tracking the rotation of its target
- bool **ShowGeneralSettings**
  INTERNAL USE ONLY
- bool **AlwaysShowCenterObject**
  Determines if the enter object or center blip should alwats be shown in th radar
- bool **CenterObjectCanScaleByDistance**
  Determines if the center object (center blip) can scale by distance
- bool **AutoScaleOnlyAtBorder**
  Determines if the center object (center blip) can scale by distance only after pasing the trcking bounds
- bool **ShowAdditionalOptions**
  INTERNAL USE ONLY
- Sprite **icon** = new Sprite()
  The blip icon if the blip is a sprite
- Sprite **TrackingLine** = new Sprite()
  The sprite which will represent the racking line

- Transform **prefab**
  prefab blip
- string **State** = ""
  INTERNAL USE ONLY
- string **Tag** = "Player"
  INTERNAL USE ONLY
- Material **SpriteMaterial**
  The material used for the sprite blip
- Color **colour**
  The colour of the sprite blip
- float **BlipSize** = 1
  The size of the blip
- float **DynamicBlipSize** = 0.05f
  The default minimum scale of the blip
- float **CenterBLipMinSize**
  The minimum scale of the blip
- float **CenterBLipMaxSize**
  The maximum Size of th eblip
- float **AutoScaleFactor** = 1
  The factor which is used to detemine by how much the blip is scales over distance
- int **Layer** = 0
- Transform **CenterBlip**
  The blip at the center of the radar
- Transform **CenterObject**
  The obje t being tracked
- **CreateBlipAs _CreateBlipAs**
  Determines what the blip should be created as , prefab or sprite

RadarDesign2D Class Reference

Public Attributes

- float **RadarDiameter** = 1
  This is the Diameter of the radar, this value will directly change the scale of the Radars child object "Designs" once UseSceneScale is false
- float **SceneScale** = 100.0f
  This is the amound of the scene that the radar is able to 'see' in order to collect dats on things to track and display
- float **TrackingBounds** = 1
  The range in which all blips can be shown in the radar
- float **InnerCullingZone** = 0f
  The diameter of the zone at the center of the radar in which all blips will ce culled

- float **RadarRotationOffset** = 0f
  INTERNAL USE ONLY
- const float **ConstantRadarRenderDistance** = 4
  Do not replace this value
- float **xPadding**
  The padding on the x and Y axis of the radar system
- **RadarPositioning radarPositioning** = RadarPositioning.Snap
  Determins if the radar will ise Manual position or Snap Positioning
- **SnapPosition snapPosition** = SnapPosition.BottomLeft
  Determines where in scren space the radar system will be positioned
- **FrontIs frontIs** = FrontIs.North
  Determining what defines the forward facing position of the radar
- Rect **RadarRect**
  INTERNAL USE ONLY
- int **BlipCount** = 0
  INTERNAL USE ONLY
- int **DesignsCount** = 0
  INTERNAL USE ONLY
- bool **UseLocalScale**
  Determines if we should use the scale of the Radar "Designs" child object instead of the RadarDiameter
- bool **Visualize** = true
  INTERNAL USE ONLY
- bool **ShowDesignSection**
  INTERNAL USE ONLY
- bool **IgnoreDiameterScale** = false
  When true, the radar ; diameter (Sale of the Radars "Designs" child object) when scales to a vlue greater or less than one will not prompt the radar system to reposition itslf automatically to maintain a correct position in screen space
- bool **LinkToTrackingBounds**
  Determines if the tracking bounds values will always be the same as
- bool **ShowRenderCameraSettings**
  INTERNAL USE ONLY
- bool **ShowScaleSettings**
  INTERNAL USE ONLY
- bool **ShowPositioningSettings**
  INTERNAL USE ONLY
- bool **ShowDesignLayers**
  INTERNAL USE ONLY
- bool **ManualCameraSetup**
  INTERNAL USE ONLY
- bool **UseMainCamera**
  determines if we will be using the gameobject in the scne with the tag "Main Camera"

- GameObject **DesignsObject**
  INTERNAL USE ONLY
- Camera **camera**
  The camera which will be the camera your player views the world through at any time
- Camera **rederingCamera**
  The camera whuch will only render radar systems, (These camera are automatically created for you)
- string **CameraTag** = "MainCamera"
  INTERNAL USE ONLY
- List< **RotationTarget** > **RotationTargets** = new List<**RotationTarget**>()
  The list of Rotation targets
- Vector3 **Pan** = new Vector3()
  The pan of the blips in the radar
